H 6

```
TTTTTTTTTTTTTTT   TTTTTTTTTTTTTTT   DDDDDDDDDDDD      RRRRRRRRRRRR      VVV         VVV   RRRRRRRRRRRR
TTTTTTTTTTTTTTT   TTTTTTTTTTTTTTT   DDDDDDDDDDDD      RRRRRRRRRRRR      VVV         VVV   RRRRRRRRRRRR
TTTTTTTTTTTTTTT   TTTTTTTTTTTTTTT   DDDDDDDDDDDD      RRRRRRRRRRRR      VVV         VVV   RRRRRRRRRRRR
      TTT               TTT         DDD         DDD   RRR         RRR   VVV         VVV   RRR         RRR
      TTT               TTT         DDD         DDD   RRR         RRR   VVV         VVV   RRR         RRR
      TTT               TTT         DDD         DDD   RRR         RRR   VVV         VVV   RRR         RRR
      TTT               TTT         DDD         DDD   RRR         RRR   VVV         VVV   RRR         RRR
      TTT               TTT         DDD         DDD   RRR         RRR   VVV         VVV   RRR         RRR
      TTT               TTT         DDD         DDD   RRR         RRR   VVV         VVV   RRR         RRR
      TTT               TTT         DDD         DDD   RRRRRRRRRRRR      VVV         VVV   RRRRRRRRRRRR
      TTT               TTT         DDD         DDD   RRRRRRRRRRRR      VVV         VVV   RRRRRRRRRRRR
      TTT               TTT         DDD         DDD   RRRRRRRRRRRR      VVV         VVV   RRRRRRRRRRRR
      TTT               TTT         DDD         DDD   RRR   RRR         VVV         VVV   RRR   RRR
      TTT               TTT         DDD         DDD   RRR   RRR         VVV         VVV   RRR   RRR
      TTT               TTT         DDD         DDD   RRR   RRR         VVV         VVV   RRR   RRR
      TTT               TTT         DDD         DDD   RRR      RRR         VVV   VVV      RRR      RRR
      TTT               TTT         DDD         DDD   RRR      RRR         VVV   VVV      RRR      RRR
      TTT               TTT         DDD         DDD   RRR      RRR         VVV   VVV      RRR      RRR
      TTT               TTT         DDDDDDDDDDDD      RRR         RRR         VVV         RRR         RRR
      TTT               TTT         DDDDDDDDDDDD      RRR         RRR         VVV         RRR         RRR
      TTT               TTT         DDDDDDDDDDDD      RRR         RRR         VVV         RRR         RRR
```

```
**FILE**ID**TTYFDT



TTTTTTTTTT  TTTTTTTTTT  YY      YY  FFFFFFFFFF  DDDDDDDD    TTTTTTTTTT
TTTTTTTTTT  TTTTTTTTTT  YY      YY  FFFFFFFFFF  DDDDDDDD    TTTTTTTTTT
    TT          TT      YY      YY  FF          DD      DD      TT
    TT          TT      YY      YY  FF          DD      DD      TT
    TT          TT        YY  YY    FF          DD      DD      TT
    TT          TT        YY  YY    FF          DD      DD      TT
    TT          TT          YY      FFFFFFFF    DD      DD      TT
    TT          TT          YY      FFFFFFFF    DD      DD      TT
    TT          TT          YY      FF          DD      DD      TT
    TT          TT          YY      FF          DD      DD      TT
    TT          TT          YY      FF          DD      DD      TT          ....
    TT          TT          YY      FF          DD      DD      TT          ....
    TT          TT          YY      FF          DDDDDDDD        TT          ....
    TT          TT          YY      FF          DDDDDDDD        TT          ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II    SS
LL              II    SS
LL              II    SS
LL              II    SS
LL              II      SSSSSS
LL              II      SSSSSS
LL              II            SS
LL              II            SS
LL              II            SS
LL              II            SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

Table of contents

```
    (2)      217          Declarations
    (2)      270          TTY$FDTREAD - FUNCTION DECISION ROUTINE FOR TERMINAL READ FUNCTIONS
    (2)      618          TTY$FDTITEMREAD - ITEM LIST SPECIFYED ON THE READ.
    (18)    1441          TTY$FDTWRITE - Function decision routine for terminal writes
    (19)    1684          TTY$FDTSETM -- FUNCTION DECISION ROUTINE FOR TERMINAL SET MODE
    (20)    1716          TTY$FDTSETC - FUNCTION DECISION ROUTINE FOR TERMINAL SET CHARS
    (40)    1959          MODE OR CHARACTERISTICS
    (41)    2055          SETMODE/CHAR service routines
    (54)    2144          MOVE TRANSLATE - TRANSLATE TO UPPERCASE
    (55)    2188          TTY$UPPER - Translate a string to upper case
    (56)    2247          TTY$FALLBACK - ROUTINE TO TRANSLATE 8-BIT CHARACTERS TO 7-BIT
```

```
0000            .TITLE  TTYFDT - Terminal driver function decision routines
0000            .IDENT  'V04-001'
0000
0000    ;
0000    ;****************************************************************************
0000    ;*                                                                          *
0000    ;*   COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                *
0000    ;*   DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                 *
0000    ;*   ALL RIGHTS RESERVED.                                                   *
0000    ;*                                                                          *
0000    ;*   THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  *
0000    ;*   ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
0000    ;*   INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER  *
0000    ;*   COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  *
0000    ;*   OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY  *
0000    ;*   TRANSFERRED.                                                           *
0000    ;*                                                                          *
0000    ;*   THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE  *
0000    ;*   AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT  *
0000    ;*   CORPORATION.                                                           *
0000    ;*                                                                          *
0000    ;*   DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS  *
0000    ;*   SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                *
0000    ;*                                                                          *
0000    ;*                                                                          *
0000    ;****************************************************************************
0000    ;
0000    ;++
0000    ; FACILITY:
0000    ;
0000    ;       VAX/VMS TERMINAL DRIVER
0000    ;
0000    ; ABSTRACT: THIS MODULE CONTAINS THE FUNCTION DECISION ROUTINES FOR TERMINAL
0000    ; RELATED I/O FUNCTIONS.
0000    ;
0000    ;       NOTE:   THIS MODULE RUNS IN THE CONTEXT OF THE LOGICAL TERMINAL
0000    ;               UCB. IT MUST NOT REFERENCE ANY PHYSICAL FIELDS WITH OUT
0000    ;               USING AN INDIRECT REFERENCE THROUGH UCB$L_TL_PHYUCB.
0000    ;               SEE THE GUIDELINES OUTLINED IN THIS MODULE.
0000    ;
0000    ; AUTHOR: R.HEINEN 23-SEPT-76 VERSION V06
0000    ;
0000    ; Revision history:
0000    ;
0000    ;       V04-001 MIR1100         Michael I. Rosenblum            06-Sep-1984
0000    ;               If read verify is specified and zero length picture string
0000    ;               was specified then the system would crash.
0000    ;
0000    ;       V03-029 MIR0450         Michael I. Rosenblum            05-Jun-1984
0000    ;               Fix problem with writes with lower clear and fallback set
0000    ;               Fix boundery problem with Readverify.
0000    ;               Fix bug in uppercase logic.
0000    ;
0000    ;       V03-028 MIR0700         Michael I. Rosenblum            25-May-1984
0000    ;               Fix bug where things would be verry screwed up if the
0000    ;               buffered i/o quota was exceded on a read.
0000    ;
```

```
0000   ;          V03-027 MIR0370          Michael I. Rosenblum              20-Mar-1984
0000   ;                  Fix bug that caused pool to be corrupted if a read with
0000   ;                  user specified terminator mask was issued when fallback
0000   ;                  was set.  Clear out new read fields.
0000   ;
0000   ;          V03-026 MIR0310          Michael I. Rosenblum              09-Feb-1984
0000   ;                  Don't give priority boosts to programs who do short I/O's
0000   ;
0000   ;          V03-025 MIR0300          Michael I. Rosenblum              30-jan-1984
0000   ;                  Made the fallback characteristic overide writeall, pasall,
0000   ;                  and pasthru.
0000   ;
0000   ;          V03-024 MIR0082          Michael I. Rosenblum              19-Aug-1983
0000   ;                  Remove CMKRNL priv check in connect.
0000   ;
0000   ;          V03-023 MIR0080          Michael I. Rosenblum              28-Jul-1983
0000   ;                  Move newline code to TTYSTRSTP
0000   ;                  Reposition routines in this module
0000   ;
0000   ;          V03-022 MIR0053          Michael I. Rosenblum              27-Jun-1983
0000   ;                  Fix bug in code that processed zero length initial strings
0000   ;                  When uppercaseing.
0000   ;
0000   ;          V03-021 MIR0051          Michael I. Rosenblum              23-Jun-1983
0000   ;                  Restructure MOVE_TRANSLATE to clean and optomize.
0000   ;                  fix bug in prompt and initial string fallback presentation
0000   ;                  dd code  to use the class relocation table for fallback
0000   ;
0000   ;          V03-020 RKS0020          RICK SPITZ                        7-JUN-1983
0000   ;                  ADD CONNECT/DISCONNECT FDT ACTION ROUTINES
0000   ;                  MOVE UCB$V_TT_HANGUP INTO LUCB.
0000   ;
0000   ;          V03-019 RKS0019          RICK SPITZ                        27-MAY-1983
0000   ;                  INTERLOCK FDT FUNCTIONS THAT REFER TO PUCB FIELDS.
0000   ;                  THIS IS NEEDED FOR CONNECT/DISCONNECT FUNCTIONALITY.
0000   ;
0000   ;                  INIT LINREST FIELD IN READ BUFFER
0000   ;
0000   ;          V03-018 MIR0049          Michael I. Rosenblum              06-May-1983
0000   ;                  Add code to handle fallback presentation of eight bit
0000   ;                  characters.
0000   ;
0000   ;          V03-017 MIR0045          Michael I. Rosenblum              05-May-1983
0000   ;                  Fix change in definition of TTY$C_FC_N_SET to
0000   ;                  TTY$C_FC_HANGUP.
0000   ;
0000   ;          V03-016 MIR0030          Michael I. Rosenblum              30-Mar-1983
0000   ;                  Integreate Read verify functionality with the normal
0000   ;                  Terminal driver as an item list read function.
0000   ;
0000   ;          V03-015 MIR0029          Michael I. Rosenblum              22-Mar-1983
0000   ;                  Add code to handle overstrike and insert modes.
0000   ;                  Fix bug in initial offset code.
0000   ;
0000   ;          V03-014 RKS0014          RICK SPITZ                        14-MAR-1983
0000   ;                  ADD SUPPORT FOR LOGICAL UCB. NOTE THAT THIS A FUNDEMENTAL
0000   ;                  CHANGE TO THE OPERATION OF THIS MODULE. WHEN RUNNING IN
```

```
0000  ;                 FDT CONTEXT, IT IS VERY IMPORTANT TO ONLY REFERENCE FIELDS
0000  ;                 IN THE LOGICAL UCB REGION. ALL REFERENCES TO PHYSICAL UCB
0000  ;                 EXTENSIONS MUST BE DONE VIA THE PHYSICAL UCB POINTER.
0000  ;
0000  ;         V03-013 MIR8026         Michael I. Rosenblum            14-Mar-1983
0000  ;                 Fix bug in initial string loading code.
0000  ;
0000  ;         V03-012 MIR7026         Michael I. Rosenblum            11-Mar-1983
0000  ;                 Copy modifier bits into read buffer modify field.
0000  ;                 Fix off by 1 error in initial offset code.
0000  ;
0000  ;         V03-011 MIR2026         Michael I. Rosenblum            07-Mar-1983
0000  ;                 Set up modifier bits in the read buffer for normal
0000  ;                 reads.
0000  ;
0000  ;         V03-010 MIR0026         Michael I. Rosenblum            01-Mar-1983
0000  ;                 Add modifier bits to turn off recall and editing
0000  ;
0000  ;         V03-009 MIR0025         Michael I. Rosenblum            01-Feb-1983
0000  ;                 Impliment alternate item list fdt routine to handle
0000  ;                 the current functionality in the itemlist QIO form.
0000  ;
0000  ;         V03-008 MIR0024         Michael I. Rosenblum            28-Jan-1983
0000  ;                 Impliment new terminal read buffer structure.
0000  ;
0000  ;         V03-007 MIR0013         Michael I. Rosenblum            16-Dec-1982
0000  ;                 Fix up refferences to new ucb structure
0000  ;
0000  ;         V03-006 MIR0011         Michael I. Rosenblum            18-Nov-1982
0000  ;                 Change CTRLR state to be EDITREAD state.
0000  ;
0000  ;         V03-005 MIR0010         Michael I. Rosenblum            09-Nov-1982
0000  ;                 Move the address of the terminator mask, and the length
0000  ;                 of the prompt string from the IRP into the terminal read
0000  ;                 packet.  Also move the count of the characters in the
0000  ;                 buffer from the UCB into the terminal typeahead buffer packet.
0000  ;
0000  ;         V03-004 RKS0004         RICK SPITZ                      23-SEP-1982
0000  ;                 INSURE CONTROL Y AS IS POSTED PRIOR ATTEMPTING TO DELIVER
0000  ;                 DEFERRED HANGUP AST. THIS CONDITION MAY OCCUR ON SLAVE
0000  ;                 TERMINALS.
0000  ;
0000  ;         V03-003 RKS0003         RICK SPITZ                      05-APR-1982
0000  ;                 ALLOW PASSALL DATA DURING UPCASE CONVERSION
0000  ;
0000  ;         V03-002 RKS0002         RICK SPITZ                      31-MAR-1982
0000  ;                 ADD SPECIAL CHARACTERISTIC BITS FOR DCL SPAWN
0000  ;                 TRANSLATE LOWER CASE OUTPUT ON UPPERCASE DEVICES.
0000  ;                 FIX SECURITY PROBLEM WITH AP AND TERMINATOR BITMAPS
0000  ;
0000  ;         V03-001 RKS0001         RICK SPITZ                      23-MAR-1982
0000  ;                 CORRECT ALTERNATE CLASS DRIVER DISPATCHING.
0000  ;                 REPAIR SECURITY PROBLEM WITH USE OF AP.
0000  ;
0000  ;         V02-035 ROW0065         Ralph O. Weber                  31-JAN-1982
0000  ;                 Move test for IO$V_EXTEND in TTY$FDTREAD so as to eliminate
0000  ;                 executing duplicate code in both the regular class driver FDT
```

```
0000 ;               and the alternate class driver FDT.  Add alternate class
0000 ;               driver legal function test before dispatching to alternate
0000 ;               class driver FDT.
0000 ;
0000 ;       V02-034 RKS034          RICK SPITZ                      24-JAN-1982
0000 ;               IRP$W_TT_PRMPT+2 ENHANCED TO SPECIFY INITIAL READ FIELD OFFSET
0000 ;
0000 ;       V02-033 RKS033          RICK SPITZ                      15-DEC-1981
0000 ;               ADD SUPPORT FOR ALTERNATE CLASS DRIVER.
0000 ;               REMOVE LOGIO REQUIREMENT FOR CONTROL Y ASTS.
0000 ;               REPAIR SET_MODEM MAINTENANCE FUNCTION.
0000 ;
0000 ;       V02-032 RKS032          RICK SPITZ                      8-NOV-1981
0000 ;               ADD OUT OF BAND SUPPORT
0000 ;
0000 ;       V02-031 JLV0101         Jake VanNoy                     27-Oct-1981
0000 ;               Changed TTYDEFS to $TTYDEFS.
0000 ;
0000 ;       V02-030 RKS030          RICK SPITZ                      15-SEP-1981
0000 ;               REDEFINE DIAGNOSTIC MODEM BIT
0000 ;
0000 ;       V02-029 RKS029          RICK SPITZ                      26-AUG-1981
0000 ;               ADD MAINT ENABLE BIT
0000 ;
0000 ;       V02-028 RKS028          RICK SPITZ                      20-AUG-1981
0000 ;               ADD SUPPORT FOR ESCAPE MODIFIER ON READ
0000 ;
0000 ;       V02-027 RKS027          RICK SPITZ                      30-APR-1981
0000 ;               THIS MODULE HAS BEEN ENHANCED TO SUPPORT QUADWORD STATE
0000 ;               AND DEVDEPEND STRUCTURES. ALSO ENHANCEMENTS WERE ADDED
0000 ;               TO SUPPORT CHANGES TO THE STRUCTURE OF THE UCB INCLUDING
0000 ;               SPLIT SPEED.
0000 ;               SUPPORT FOR DIAGNOSTIC MAINTENANCE FUNCTIONS HAS BEEN ADDED
0000 ;               AS WELL AS NEW FIELDS IN THE TWP. THIS ALLOWS FORKING
0000 ;               ON THE TWP TO ALLOW ALLOCATION/DEALLOCATION OF MAP REGISTERS.
0000 ;
0000 ;       V02-026 RKS026          RICK SPITZ                      26-FEB-1981
0000 ;               DELETE V2.0 AUDIT TRAIL
0000 ;
0000 ;       V02-025 SPF0001         Steve Forgey                    19-Dec-1980
0000 ;               Add RTE prompt support.
0000 ;
0000 ;
0000 ;--
```

```
                0000             .SBTTL  Declarations
                0000
                0000    ;
                0000    ; EXTERNAL SYMBOLS:
                0000    ;
                0000             $ACBDEF                              ; DEFINE AST CONTROL BLOCK
                0000             $ARBDEF                              ; DEFINE ACCESS RIGHTS BLOCK
                0000    ;        $CADEF                               ; DEFINE CONDITIONAL ASSEMBLY PARAMETERS
                0000             $DDTDEF                              ; DEFINE DDT OFFSETS
                0000             $DYNDEF                              ; DEFINE DYNAMIC MEMORY BLOCK TYPES
                0000             $IODEF                               ; DEFINE I/O FUNCTION CODES
                0000             $IPLDEF                              ; DEFINE IPL CONSTANTS
                0000             $IRPDEF                              ; DEFINE I/O PACKET OFFSETS
                0000             $JIBDEF                              ; DEFINE JIB OFFSETS
                0000             $PCBDEF                              ; DEFINE PCB OFFSETS
                0000             $PRVDEF                              ; DEFINE PRIVILEGES
                0000             $PSLDEF                              ; DEFINE PSL OFFSETS
                0000             $SSDEF                               ; Define system status codes
                0000             $UCBDEF                              ; DEFINE UCB
                0000             $TRMDEF                              ; define bits for itemlist
                0000             $TTYDEF                              ; DEFINE TERMINAL DRIVER SYMBOLS
                0000             $TTDEF                               ; DEFINE TERMINAL CHARACTERISTICS
                0000             $TT2DEF                              ; EXTENDED TERMINAL CHARACTERISTICS
                0000             $TTYMACS                             ; TERMINAL MACROS
                0000             $TTYDEFS                             ; DEFINE TERMINAL DEFINITIONS
                0000    ;
                0000    ; LOCAL DEFINITIONS
                0000    ;
                0000    ; QIO ARGUMENT LIST OFFSETS
                0000    ;
    00000000    0000    P1 = 0
    00000004    0000    P2 = 4
    00000008    0000    P3 = 8
    0000000C    0000    P4 = 12
    00000010    0000    P5 = 16
    00000014    0000    P6 = 20
                0000    ;
                0000    ; make sure that definitions of the terminal item list modifiers match
                0000    ; system IO modifer definitions
                0000    ;
                0000    ASSUME  TRM$V_TM_NOECHO EQ IO$V_NOECHO                       ;NOECHO
                0000    ASSUME  TRM$V_TM_TIMED EQ IO$V_TIMED                         ;TIMED
                0000    ASSUME  TRM$V_TM_CVTLOW EQ IO$V_CVTLOW                       ;CONVERT LOWER CASE
                0000    ASSUME  TRM$V_TM_NOFILTR EQ IO$V_NOFILTR                                 ;NO FILTER
                0000    ASSUME  TRM$V_TM_DSABLMBX EQ IO$V_DSABLMBX                               ;DISABLE MAI
                0000    ASSUME  TRM$V_TM_PURGE EQ IO$V_PURGE                         ;PURGE TYPEAHEAD
                0000    ASSUME  TRM$V_TM_TRMNOECHO EQ IO$V_TRMNOECHO                 ;TERMINATORS ARE NOT
                0000    ASSUME  TRM$V_TM_REFRESH EQ IO$V_REFRESH                                 ;Control-R i
                0000    ASSUME  TRM$V_TM_ESCAPE EQ IO$V_ESCAPE                       ;TERMINATE READ ON E
                0000
                0000
```

```
00000000          .PSECT  $$$115_DRIVER,LONG
    0000          .SBTTL  TTY$FDTREAD - FUNCTION DECISION ROUTINE FOR TERMINAL READ FUNCTIONS
    0000    ;++
    0000    ; TTY$FDTREAD - FUNCTION DECISION ROUTINE FOR TERMINAL READ
    0000    ;
    0000    ; FUNCTIONAL DESCRIPTION:
    0000    ;
    0000    ; THIS ROUTINE IS THE FUNCTION DECISION ACTION ROUTINE FOR TERMINAL READS.
    0000    ;
    0000    ; THE TERMINAL READ QIO PARAMETERS ARE:
    0000    ;
    0000    ;       P1 = ADDRESS OF THE BUFFER TO RECEIVE THE DATA RECORD
    0000    ;       P2 = SIZE OF THE P1 BUFFER
    0000    ;       P3 = NUMBER OF SECONDS TO WAIT FOR CHARACTERS (IO$M_TIMED ONLY)
    0000    ;       P4 = ADDRESS OF TERMINATOR CLASS BITMASK OR 0 IF STANDARD
    0000    ;       P5 = ADDRESS OF PROMPT STRING FOR IO$_READPROMPT
    0000    ;       P6 = SIZE OF PROMPT STRINT FOR IO$_READPROMPT
    0000    ;
    0000    ; THE FUNCTION PARAMETERS ARE VALIDATED AND IF CORRECT, THE PACKET IS
    0000    ; QUEUED ON THE UNIT I/O QUEUE.
    0000    ; THE PACKET CONTAINS THE FOLLOWING:
    0000    ;
    0000    ;       1. IRP$Q_TT_STATE IS SET UP TO BE THE NEW TERMINAL STATES AT THE
    0000    ;          TIME THE READ OPERATION IS STARTED.
    0000    ;       2. IRP$L_SVAPTE CONTAINS THE ADDRESS OF THE READ BUFFER
    0000    ;          FORMATTED AS FOLLOWS.
    0000    ;
    0000    ;               .LONG   ADDRESS TO STORE DATA
    0000    ;               .LONG   USER BUFFER VIRTUAL ADDRESS
    0000    ;               .WORD   SIZE
    0000    ;               .WORD   TYPE
    0000    ;               .WORD   STORAGE FOR STARTING CURSOR POSITION
    0000    ;               .WORD TIMEOUT COUNT
    0000    ;               PROMPT STRING
    0000    ;               READ BUFFER
    0000    ;               TERMINATOR MASK FOR NONSTANDARD CLASSES
    0000    ;
    0000    ;       3. IRP$L_TT_TERM ADDRESSES THE TERMINATOR BITMASK
    0000    ;       4. IRP$W_FUNC<0:6> ARE SET FOR A FAST CASE ON FUNCTION TYPE
    0000    ;       5. IRP$W_BOFF IS THE QUOTA FOR THE I/O
    0000    ;       6. IRP$W_BCNT IS THE READ REQUEST SIZE
    0000    ;
    0000    ; STATE BIT USAGE.
    0000    ;
    0000    ;       FOR IO$_READPBLK, TTY$V_ST_PASSALL IS SET.
    0000    ;       FOR IO$_READPROMPT, TTY$V_ST_PROMPT AND TTY$V_ST_EDITREAD ARE SET.
    0000    ;               EDITREAD WILL FORCE THE PROMPT AND INITIAL STRING OUT.
    0000    ;
    0000    ;       FOR IO$M_NOECHO, TTY$V_ST_NOECHO IS SET.
    0000    ;       FOR IO$M_NOFILTR, TTY$V_ST_NOFILTR IS SET.
    0000    ;       For IO$M_REFRESH, TTY$V_ST_REFRSH is set.
    0000    ;
    0000    ; INPUTS:
    0000    ;
    0000    ;       R3 = ADDRESS OF THE PACKET FOR THIS REQUEST
    0000    ;       R4 = CURRENT PCB
    0000    ;       R5 = UCB ADDRESS
```

```
                                0000   326 ;           R6 = ASSIGNED CCB
                                0000   327 ;           R7 = FUNCTION CODE
                                0000   328 ;           AP = ADDRESS OF FIRST FUNCTION DEPENDENT QIO PARAMETER
                                0000   329 ;
                                0000   330 ; OUTPUTS:
                                0000   331 ;
                                0000   332 ;           THE I/O IF IN ERROR IS COMPLETED VIA "EXE$ABORTIO".
                                0000   333 ;           THE I/O IF VALID IS QUEUED TO THE DRIVER BY "EXE$QIODRVPKT".
                                0000   334 ;
                                0000   335 ; COMPLETION CODES:
                                0000   336 ;
                                0000   337 ;           SS$_ACCVIO - ACCESS VIOLATION ON BUFFER.
                                0000   338 ;           SS$_EXQUOTA - OVER QUOTA FOR BUFFERED I/O
                                0000   339 ;           SS$_INSFMEM - INSUFFICIENT MEMORY
                                0000   340 ;--
                                0000   341             .ENABLE LSB
                                0000   342 TTY$FDTREAD::                                      ;
                   0F     E1    0000   343             BBC     #IO$V_EXTEND,-
             03 20 A3           0002   344                     IRP$W_FUNC(R3),2$
                 0262     31    0005   345             BRW     TTY$FDTITEMREAD
                                0008   346 2$:
                   07     E0    0008   347             BBS     #TT$V_LOWER,-                   ; IS THIS LOWER CASE
             05 44 A5           000A   348                     UCB$L_DEVDEPEND(R5),1$          ; NO THEN CONTINUE ON
                   08     E2    000D   349             BBSS    #IO$V_CVTLOW,-
             00 20 A3           000F   350                     IRP$W_FUNC(R3),1$               ; YES THEN SET LOWER IN FUNCTION
                   53     DD    0012   351 1$:         PUSHL   R3                              ; SAVE PACKET ADDRESS
                                0014   352 ;
                                0014   353 ; SET PROPER STATE BITS FOR READ FUNCTIONS FROM FUNCTION MODIFIERS
                                0014   354 ;
                                0014   355 ; NOTE THE CORRESPONDENCE OF THE VALUES
                                0014   356 ;
                          78    0014   357             ASHL    #TTY$V_ST_NOECHO-               ; Move function code and its
                                0015   358                     -IO$V_NOECHO,-                  ; modifiers into bits 9-25 of
   58    20 A3     FD 8F        0015   359                     IRP$W_FUNC(R3),R8               ; a register.
     58    FFFFF3B7 8F    CA    001A   360             BICL    #^C<TTY$M_ST_NOECHO!-           ; Clear all bits except NOECHO
                                0021   361                     TTY$M_ST_NOFLTR!-               ; NOFLTR, and
                                0021   362                     TTY$M_ST_ESCAPE!-               ; ESCAPE
                                0021   363                     TTY$M_ST_REFRSH>,R8             ; REFRESH if specified.
       1A 48 A5    0C     E1    0021   364             BBC     #TT2$V_EDITING,UCB$L_DEVDEPND2(R5),3$; IS THIS AN EDITING READ
     58    00100000 8F    C8    0026   365             BISL    #TTY$M_ST_EDITING,R8            ; YES THEN SET EDITING
     58    00800000 8F    C8    002D   366             BISL    #TTY$M_ST_OVERSTRIKE,R8         ; THEN MAKE OVERSTRIKE THE DEFAULT
       07 48 A5    0D     E1    0034   367             BBC     #TT2$V_INSERT,UCB$L_DEVDEPND2(R5),3$; IF HE WANTS INSERT THEN
     58    00800000 8F    CA    0039   368             BICL    #TTY$M_ST_OVERSTRIKE,R8         ; CLEAR INSERT
              1000 8F     3C    0040   369 3$:         MOVZWL  #TTY$M_ST_READ,-
                40 A3           0044   370                     IRP$Q_TT_STATE(R3)              ; INIT AND ADD READ
                                0046   371 ;
                                0046   372 ; CHECK ACCESS TO READ BUFFER
                                0046   373 ;
              50    6C    D0    0046   374             MOVL    P1(AP),R0                       ; GET BUFFER ADDRESS AND SIZE
         51    04 AC      3C    0049   375             MOVZWL  P2(AP),R1                       ;
                   05     12    004D   376             BNEQ    5$                              ; IF NEQ THEN ACTUAL READ
                   08     E3    004F   377             BBCS    #TTY$V_ST_EOL,-
             06 40 A3           0051   378                     IRP$Q_TT_STATE(R3),10$          ; SET EOL AND BRANCH
         00000000'GF      16    0054   379 5$:         JSB     G^EXE$READCHK                   ; CHECK READ ACCESS FOR BUFFER
                                005A   380                                                     ; NO RETURN MEANS NO ACCESS
              59    50    7D    005A   381 10$:        MOVQ    R0,R9                           ; COPY INPUT BUFFER PARAMS
                    57    D4    005D   382             CLRL    R7                              ; ASSUME 0 BUFFER SIZE
```

```
0C    20 A3    06    00    ED  005F           CMPZV   #IRP$V_FCODE,#IRP$S_FCODE,IRP$W_FUNC(R3),#IO$_READPBLK; PASSALL?
                     08    13  0065           BEQL    12$                           ; IF EQL THEN YES
               06    00    ED  0067           CMPZV   #IRP$V_FCODE,#IRP$S_FCODE,-      ; TEMP READ PASSALL
         3A    20    A3        006A                   IRP$W_FUNC(R3),#IO$_TTYREADALL
                     04    12  006D           BNEQ    20$                           ; NO, BRANCH
                               006F      12$:
                               006F
                               006F           .IF DF CA$_MEASURE_IOT
                               006F
                               006F           BLBC    G^PMS$GL_DOSTATS,15$          ; IF FLAG OFF BYPASS NEXT INST.
                               006F           INCL    G^PMS$GL_PASSALL              ; ELSE INCR PASSALL COUNTER
                               006F
                               006F           .ENDC
                               006F
         65 58       02    E3  006F      15$: BBCS    #TTY$V_ST_PASALL,R8,50$ ; SET PASSALL MODE AND BR
                               0073      ;
                               0073      ; DO SPECIAL FUNCTION LOGIC FOR READ WITH PROMPT
                               0073      ;
               06    00    ED  0073      20$: CMPZV   #IRP$V_FCODE,#IRP$S_FCODE,-      ; READPROMPT?
         37    20    A3        0076                   IRP$W_FUNC(R3),#IO$_READPROMPT
                     0C    13  0079           BEQL    22$                           ; YES, BRANCH
               06    00    ED  007B           CMPZV   #IRP$V_FCODE,#IRP$S_FCODE,-      ; READ PASSALL W/PROMPT?
         3B    20    A3        007E                   IRP$W_FUNC(R3),#IO$_TTYREADPALL
                     55    12  0081           BNEQ    50$                           ; BRANCH IF NO
         00 58       02    E3  0083           BBCS    #TTY$V_ST_PASALL,R8,22$ ; SET PASSALL BIT IN VECTOR
                               0087      22$:
                               0087      ;
                               0087      ; SEE IF NO PROMPT IS SPECIFIED
                               0087      ;
         57    14    AC    3C  0087           MOVZWL  P6(AP),R7                     ; GET SIZE OF PROMPT
                     4B    13  008B           BEQL    50$                           ; IF EQL THEN MAKE THIS NORMAL READ
                               008D      ;
                               008D      ; READ WITH PROMPT
                               008D      ;
                               008D
                               008D           .IF DF CA$_MEASURE_IOT
                               008D
                               008D           BLBC    G^PMS$GL_DOSTATS,30$          ; IF FLAG OFF, BYPASS NEXT INST
                               008D           INCL    G^PMS$GL_RWP                  ; INCR READ WITH PROMPT COUNTER
                               008D           CMPL    #12,R7                        ; \ISOLATE READ WITH PROMPTS
                               008D           BGTR    25$                           ; /GREATER THAN 12 CHARS
                               008D           INCL    G^PMS$GL_LRGRWP               ; INCR CTR FOR PROMPTS > 12 CHARS
                               008D      25$: ADDL2   R7,G^PMS$GL_RWPSUM            ; KEEP RUNNING SUM OF RWP SIZES
                               008D
                               008D           .ENDC
                               008D
            58       20    C8  008D      30$: BISL    #TTY$M_ST_PROMPT,R8           ; INSERT BITS FOR PROMPT
40 A3    00000200    8F    C8  0090           BISL    #TTY$M_ST_EDITREAD,IRP$Q_TT_STATE(R3); INSERT BITS FOR PROMPT
            50    10 AC    D0  0098           MOVL    P5(AP),R0                     ; GET PROMPT BUFFER ADDRESS
                               009C      ;
                               009C      ; CHECK ACCESS TO PROMPT STRING
                               009C      ;
            51       57    3C  009C           MOVZWL  R7,R1                         ; GET SIZE PROMPT
         00000000'GF       16  009F           JSB     G^EXE$WRITECHK                ; CHECK PROMPT BUFFER ACCESS
                               00A5                                                 ; NO RETURN MEANS NO ACCESS
                  4E A3    B4  00A5           CLRW    IRP$W_TT_PRMPT+2(R3)          ; ZERO INITIAL READ OFFSET
                     0E    E1  00A8           BBC     #TT2$V_FALLBACK,-
```

```
          1C 48 A5        00AA   440                      UCB$L_DEVDEPND2(R5),35$ ; ARE WE DOING FALLBACK?
                59    DD  00AD   441             PUSHL    R9                      ; SAVE THIS SPECAIL
                0F    BB  00AF   442             PUSHR    #^M<R0,R1,R2,R3>        ; SAVE A REGISTER
            52  50    D0  00B1   443             MOVL     R0,R2                   ; REGISTERS NEED TO BE SWAPPED
            50  51    D0  00B4   444             MOVL     R1,R0                   ;...
            51  52    D0  00B7   445             MOVL     R2,R1                   ;
              0CA9    30  00BA   446             BSBW     ADDFALL                 ; GET THE NUMBER OF CHARACTERS
                0F    BA  00BD   447             POPR     #^M<R0,R1,R2,R3>
         4E A3  59    B0  00BF   448             MOVW     R9,IRP$W_TT_PRMPT+2(R3) ; SAVE THE ADDITIONAL NUMBER OF CHARCTERS
            51  59    C0  00C3   449             ADDL     R9,R1                   ; AND FIX UP R1
                59  8ED0  00C6   450             POPL     R9                      ; RESTORE R9
            56  50    D0  00C9   451 35$:        MOVL     R0,R6                   ; COPY PROMPT STRING ADDRESS
            51  5A    C0  00CC   452             ADDL     R10,R1                  ; ADJUST PROMPT SIZE TO OVERALLOCATE
         32 A3  5A    B0  00CF   453             MOVW     R10,IRP$W_BCNT(R3)      ; SAVE ACTUAL READ SIZE
      00 2A A3  01    E2  00D3   454             BBSS     #IRP$V_FUNC,IRP$W_STS(R3),50$; RESET TRANSFER DIRECTION
         4C A3  57    B0  00D8   455 50$:        MOVW     R7,IRP$W_TT_PRMPT(R3)   ; keep the prompt size
   51     0000004A 8F    C0  00DC   456             ADDL     #TTY$L_RB_DATA,R1       ; ADJUST SIZE FOR BLOCK HEADER
      15 44 A5  06    E1  00E3   457             BBC      #TT$V_SCRIPT,UCB$L_DEVDEPEND(R5),55$; RTE TERMINAL LINE ?
            4C  A3    B6  00E8   458             INCW     IRP$W_TT_PRMPT(R3)      ; ADD RTE PROMPT SIZE
                51    D6  00EB   459             INCL     R1                      ; ADD RTE PROMPT SIZE
            58  20    C8  00ED   460             BISL2    #TTY$M_ST_PROMPT,R8     ; ENSURE FUNCTION IS A RWP
       00000200 8F    C8  00F0   461             BISL2    #TTY$M_ST_EDITREAD,-
                40 A3     00F6   462                      IRP$Q_TT_STATE(R3)      ; ENSURE FUNCTION IS A RWP
      00 2A A3  01    E2  00F8   463             BBSS     #IRP$V_FUNC,IRP$W_STS(R3),55$; ENSURE TRANFER DIRECTION RESET
3C A3    00000000'EF  9E  00FD   464 55$:        MOVAB    TTY$A_STANDARD,IRP$L_TT_TERM(R3); ASSUME STANDARD TERMINATORS
   38 A3     0C AC    D0  0105   465             MOVL     P4(AP),IRP$L_MEDIA(R3)  ; GET ADDRESS OF TERMINATOR BITMASK
                2B    13  010A   466             BEQL     65$                     ; IF EQL THEN STANDARD
      52     38 A3    D0  010C   467             MOVL     IRP$L_MEDIA(R3),R2      ; RETRIEVE ADDRESS
         3C A3  51    3C  0110   468             MOVZWL   R1,IRP$L_TT_TERM(R3)    ; SAVE OFFSET IN BUFFER BLOCK
            50  0C    3C  0114   469             MOVZWL   #SS$_ACCVIO,R0          ; ASSUME ACCESS VIOLATION
                          0117   470             IFNORD   #8,(R2),70$             ; DESC. ACCESSIBLE?
      5B     04 A2    D0  011D   471             MOVL     4(R2),R11               ; GET BITMAP ADDRESS
            5A  62    3C  0121   472             MOVZWL   (R2),R10                ; GET BITMAP SIZE
                0E    13  0124   473             BEQL     60$                     ; IF EQL THEN SHORT FORM
            20  5A    B1  0126   474             CMPW     R10,#32                 ; CHECK FOR VALID LENGTH
                03    1B  0129   475             BLEQU    57$                     ; RANGE OK
            5A  20    3C  012B   476             MOVZWL   #32,R10                 ; USE MAXIMUM
                          012E   477 57$:        IFNORD   R10,(R11),70$           ; BITMAP ACCESSIBLE?
            51  20    C0  0134   478 60$:        ADDL     #32,R1                  ; ASSUME EIGHTBIT CHARACTERS AND EXTEND
                          0137   479                                              ; TERMINATOR MASK
                          0137   480 ;
                          0137   481 ; ALLOCATE HEADER + PROMPT + DATA + BITMASK
                          0137   482 ;
                          0137   483 ; CHECK FOR BUFFERED I/O QUOTA
                          0137   484 ;
                          0137   485 65$:
       00000000'GF    16  0137   486             JSB      G^EXE$BUFFRQUOTA        ; CHECK QUOTA
             06 50    E9  013D   487             BLBC     R0,70$                  ; SIGNAL ERROR IF LOW CLEAR
                          0140   488 ;
                          0140   489 ; ALLOCATE THE BUFFER
                          0140   490 ;
       00000000'GF    16  0140   491             JSB      G^EXE$ALLOCBUF          ; ALLOCATE THE BUFFER
                53  8ED0  0146   492 70$:        POPL     R3                      ; RESTORE PACKET
             03 50    E8  0149   493             BLBS     R0,72$                  ; CONTINUE IF NO ERROR
              00DF    31  014C   494             BRW      100$                    ; IF ERROR, THEN ABORT I/O
                          014F   495 ;
                          014F   496 ; FALLBACK PRESENTATION FOR PROMPTS
```

```
                                014F   497 ;
                                014F   498 72$:                                          ;
            0E 44 A5   06   E1  014F   499              BBC     #TT$V_SCRIPT,UCB$L_DEVDEPEND(R5),74$; RTE SCRIPT LINE ?
               50   4A A2   9E  0154   500              MOVAB   TTY$L_RB_DATA(R2),R0     ; GET ADDRESS FOR RTE PROMPT CHARACTER
                    50 57   C0  0158   501              ADDL2   R7,R0
         60   00000000'9F   90  015B   502              MOVB    @#PMS$GB_PROMPT,(R0)     ; INSERT RTE PROMPT
               2C A3   52   D0  0162   503 74$:         MOVL    R2,IRP$L_SVAPTE(R3)      ; SAVE ADDRESS OF BUFFERED I/O BLOCK
               30 A3   51   B0  0166   504              MOVW    R1,IRP$W_BOFF(R3)        ; SAVE BLOCK SIZE AS QUOTA
                                016A   505 ;
                                016A   506 ; ADJUST BUFFERED I/O QUOTA
                                016A   507 ;
            54    0080 C4   D0  016A   508              MOVL    PCB$L_JIB(R4),R4         ; GET JIB ADDRESS
                    51 51   3C  016F   509              MOVZWL  R1,R1                    ; CONVERT TO LONG WORD COUNT
              20 A4    51   C2  0172   510              SUBL    R1,JIB$L_BYTCNT(R4)      ; ADJUST QUOTA WORD
                                0176   511 ;
                                0176   512 ;            MARK PACKET AS TERMINAL I/O
                                0176   513 ;
         2A A3    0200 8F   A8  0176   514              BISW    #IRP$M_TERMIO,IRP$W_STS(R3)      ; SET FLAG FOR GREATER PRIORITY
                                017C   515                                               ; INCREMENTS UPON COMPLETION
                                017C   516 ;
                                017C   517 ; SET UP BUFFERED I/O BLOCK
                                017C   518 ;
           20 A2   20 A3    3C  017C   519              MOVZWL  IRP$W_FUNC(R3),TTY$L_RB_MOD(R2); SETUP THE MODIFIER WORD
      20 A2   06   00   00  F0  0181   520              INSV    #0,#IRP$V_FCODE,#IRP$S_FCODE,TTY$L_RB_MOD(R2); CLEAN
                                0187   521                                               ; OUT THE FUNCTION CODE BITS
              04 A2    59   D0  0187   522              MOVL    R9,TTY$L_RB_UVA(R2)      ; INSERT USER VIRTUAL ADDRESS
               62   4A A2   9E  018B   523              MOVAB   TTY$L_RB_DATA(R2),TTY$L_RB_TXT(R2); INSERT POINTER TO DATA AREA
                       2C   BB  018F   524              PUSHR   #^M<R2,R3,R5>            ; SAVE REGISTERS
           1C A2   3C A3    D0  0191   525              MOVL    IRP$L_TT_TERM(R3),TTY$L_RB_TERM(R2); MOVE THIS DATA INTO THE READ BU
                   4C A3    A1  0196   526              ADDW3   IRP$W_TT_PRMPT(R3),-
                   4E A3        0199   527                      IRP$W_TT_PRMPT+2(R3),-
                   34 A2        019B   528                      TTY$W_RB_PRMLEN(R2)      ; GET THE PROMPT LENGTH
           40 A2   32 A3    B0  019D   529              MOVW    IRP$W_BCNT(R3),TTY$W_RB_TXTSIZ(R2); THE TEXT SIZE
              55   34 A2    3C  01A2   530              MOVZWL  TTY$W_RB_PRMLEN(R2),R5   ;GET THE PROMPT LENGTH
                   62  55   C0  01A6   531              ADDL    R5,TTY$L_RB_TXT(R2)      ; SETUP THE OFFSET'S RIGHT
              2C A2    62   D0  01A9   532              MOVL    TTY$L_RB_TXT(R2),TTY$L_RB_LIN(R2); SETUP THE LINE ADDRESS ALSO
                   30 A2    B4  01AD   533              CLRW    TTY$W_RB_LINOFF(R2)      ; START THE OFFSETS AT ZERO
                   3C A2    B4  01B0   534              CLRW    TTY$W_RB_TXTOFF(R2)      ; ...
                   32 A2    B4  01B3   535              CLRW    TTY$W_RB_LINREST(R2)     ; AND OTHER FIELDS TO BE ZERO
                   2A A2    B4  01B6   536              CLRW    TTY$W_RB_RDSTATE(R2)
                   44 A2    B4  01B9   537              CLRW    TTY$W_RB_MODE(R2)        ; ZERO OUT THE MODE FIELD ALSO
                   38 A2    B4  01BC   538              CLRW    TTY$W_RB_CPZCUR(R2)      ; CLEAN THE CURSOR POSITION
                   14 A2    D4  01BF   539              CLRL    TTY$L_RB_ECHSTR(R2)      ; CLEAN THE ECHO STRING
              3C A3    01   CE  01C2   540              MNEGL   #1,IRP$L_TT_TERM(R3)     ;***to be removed (just for insurance)
              4C A3    01   AE  01C6   541              MNEGW   #1,IRP$W_TT_PRMPT(R3)    ;***to be removed (just for insurance)
              54   38 A3    D0  01CA   542              MOVL    IRP$L_MEDIA(R3),R4       ; ADDRESS BITMAP ADDRESS
                       2A   13  01CE   543              BEQL    90$                      ; IF EQL THEN STANDARD
                                01D0   544
                                01D0   545              .IF DF CA$_MEASURE_IOT
                                01D0   546
                                01D0   547              BLBC    G^PMS$GL_DOSTATS,75$     ; IF FLAG OFF, BYPASS NEXT INSTRUCTION
                                01D0   548              INCL    G^PMS$GL_NOSTDTRM        ; ELSE, INCR NON STD. TERMINATOR CTR
                                01D0   549              TSTL    P6(AP)                   ; \CHECK IF PROMPT SIZE
                                01D0   550              BEQL    75$                      ; /= 0
                                01D0   551              INCL    G^PMS$GL_RWPNOSTD        ; INCR CTR OF RWP WITH NON STD TERMS
                                01D0   552
                                01D0   553              .ENDC
```

```
                               01D0
             1C A2   52   C0   01D0   75$:     ADDL    R2,TTY$L_RB_TERM(R2)       ; CALC ADDR OF BITMAP SPACE
             51   34 A2   A2   01D4            SUBW    TTY$W_RB_PRMLEN(R2),R1     ; CALC BITMAP SIZE
             51   40 A2   A2   01D8            SUBW    TTY$W_RB_TXTSIZ(R2),R1     ;
          51    004A 8F   A2   01DC            SUBW    #TTY$C_RB_DATA,R1          ; AND REMOVE THE LENGTH OF THE PACKET DATA
                     5A   B5   01E1            TSTW    R10                        ; CHECK SIZE OF MAP
                     07   12   01E3            BNEQ    80$                        ; IF NEQ THEN SPECIFIED
                5A   04   9A   01E5            MOVZBL  #4,R10                     ; SET SIZE OF SHORT FORM
             5B   04 A4   9E   01E8            MOVAB   4(R4),R11                  ; ADDRESS SHORT FORM LONG WORD
   1C B2   51   00   6B   5A   2C   01EC   80$:     MOVC5   R10,(R11),#0,R1,@TTY$L_RB_TERM(R2); COPY BITMAP
          58   01000000 8F   C8   01F3            BISL    #TTY$M_ST_TERMNORM,R8      ; IF THE USER SPECIFIES THE
                               01FA                                               ; TERMINATOR MASK THEN LET HIM HAVE IT
                52   8E   D0   01FA   90$:     MOVL    (SP)+,R2                   ; RETRIEVE BLOCK ADDRESS
                53   6E   D0   01FD            MOVL    (SP),R3                    ; RETRIEVE THE IRP ADDRESS ALSO
          36 A2   08 AC   B0   0200            MOVW    P3(AP),TTY$W_RB_TIMOS(R2); GET TIMEOUT COUNT
                               0205   ;
                               0205   ; READ WITH PROMPT?
                               0205   ;
             13 58   05   E1   0205            BBC     #TTY$V_ST_PROMPT,R8,95$    ; BR IF NOT PROMPT READ
             55   04 AE   D0   0209            MOVL    4(SP),R5                   ; GET UCB ADDRESS
                     07   E1   020D            BBC     #TT$V_LOWER,-              ; IS THIS LOWER CASE
                22 44 A5       020F                    UCB$L_DEVDEPEND(R5),150$; NO THEN CONTINUE ON
                     0E   E0   0212            BBS     #TT2$V_FALLBACK,-
                1D 48 A5       0214                    UCB$L_DEVDEPND2(R5),150$; BRANCH IF FALLBACK
                               0217   93$:
        4A A2   66   57   28   0217            MOVC3   R7,(R6),TTY$L_RB_DATA(R2); COPY PROMPT STRING TO BUFFERED BLOCK
                               021C
                     28   BA   021C   95$:     POPR    #^M<R3,R5>                 ; RESTORE IRP ADDRESS
                               021E   ;
                               021E   ; QUEUE I/O PACKET TO UNIT
                               021E   ;
             44 A3   58   D0   021E            MOVL    R8,IRP$Q_TT_STATE+4(R3)    ; PUT STATE DATA IN PACKET
   20 A3   06   00   00   F0   0222            INSV    #TTY$C_FC_READ,#IRP$V_FCODE,#IRP$S_FCODE,IRP$W_FUNC(R3)
          00000000'GF   17   0228            JMP     G^EXE$QIODRVPKT            ; INSERT PACKET ON QUEUE
                               022E   ;
                               022E   ; ERROR IN PROCESSING
                               022E   ;
          00000000'GF   17   022E   100$:    JMP     G^EXE$ABORTIO              ; THE I/O IS DONE
                               0234
                               0234   ; TRANSLATE TO UPPERCASE
                               0234
                               0234   150$:
                     0E   E0   0234            BBS     #TT2$V_FALLBACK,-
                0E 48 A5       0236                    UCB$L_DEVDEPND2(R5),160$; BRANCH IF FALLBACK
             DA 58   02   E0   0239            BBS     #TTY$V_ST_PASALL,R8,93$    ; NO TRANSLATE IF PASSALL
                     00   E0   023D            BBS     #TT$V_PASSALL,-
                D5 44 A5       023F                    UCB$L_DEVDEPEND(R5),93$    ; CHECK FOR PASSALL OR PASTHRU
                     12   E0   0242            BBS     #TT2$V_PASTHRU,-
                D0 48 A5       0244                    UCB$L_DEVDEPND2(R5),93$    ; ...
                     58   DD   0247   160$:    PUSHL   R8                         ; SAVE R8
                     58   D4   0249            CLRL    R8                         ; CLEAR FLAG
                     07   E0   024B            BBS     #TT$V_LOWER,-              ; IS THIS LOWER CASE
                03 44 A5       024D                    UCB$L_DEVDEPEND(R5),165$; NO THEN CONTINUE ON
                58   01   D0   0250            MOVL    #1,R8                      ; SET DO LOWER CASE BIT
             51   4E A3   3C   0253   165$:    MOVZWL  IRP$W_TT_PRMPT+2(R3),R1    ; GET ADDITIONAL LENGTH
                  4E A3   B4   0257            CLRW    IRP$W_TT_PRMPT+2(R3)       ; CLEAN THE INITIAL OFFSET OUT
             53   57   51   C1   025A            ADDL3   R1,R7,R3                   ; AND ADD IN THE LENGTH
```

```
                 52   4A A2    9E  025E   611          MOVAB   TTY$L_RB_DATA(R2),R2     ; GET DESTINATION ADDRESS
                        0A3E   30  0262   612          BSBW    MOVE_TRANSLATE           ; TRANSLATE DATA
                           58 8ED0  0265   613          POPL    R8
                           B2   11  0268   614          BRB     95$
                                    026A   615
```

```
026A   .DISABLE LSB
026A          .sbttl  TTY$FDTITEMREAD - ITEM LIST SPECIFYED ON THE READ.
026A   ;++
026A   ; TTY$FDTITEMREAD - FUNCTION DECISION ROUTINE FOR TERMINAL READS WITH
026A   ;        ITEM LISTS (EXTEND MODE).
026A   ;
026A   ; FUNCTIONAL DESCRIPTION:
026A   ;
026A   ;  THE TERMINAL READ QIO PARAMETERS ARE:
026A   ;
026A   ;       P1 = ADDRESS OF THE BUFFER TO RECIEVE THE DATA RECORD
026A   ;       P2 = SIZE OF P1 BUFFER
026A   ;       P3 = ACCESS MODE AT WHICH THE ITEM LIST IS TO BE PROBED(OPTIONAL)
026A   ;       P4 = MBZ
026A   ;       P5 = ADDRESS OF THE ITEM LIST BUFFER
026A   ;       P6 = LENGTH IN BYTES OF THE ITEM LIST BUFFER
026A   ;
026A   ; THE ITEM LIST BUFFER IS PROBED AND THEN EACH ITEM IS VALIDATED.
026A   ; IF ALL THE ITEMS ARE CORRECT THEN THE READ PACKET IS QUEUED TO THE
026A   ; I/O QUEUE.
026A   ;
026A   ; THE PACKET CONTAINS THE FOLLOWING:
026A   ;
026A   ;       1.  IRP$Q_TT_STATE - SETUP TO BE THE NEW TERMINAL STATES AT THE
026A   ;               TIME THE READ IS STARTED.
026A   ;
026A   ;       2.  IRP$L_SVAPTE - CONTAINS THE ADDRESS OF THE READ BUFFER FORMATTED
026A   ;               AS FOLLOWS:
026A   ;       +--------------------------------------------------------------+
026A   ;       !                         TTY$L_RB_TXT                         !
026A   ;       +--------------------------------------------------------------+
026A   ;       !                         TTY$L_RB_UVA                         !
026A   ;       +--------------------------------------------------------------+
026A   ;       !TTY$B_RB_ECHLEN!    Spare      !          TTY$W_RB_SIZE       !
026A   ;       +--------------------------------------------------------------+
026A   ;       !                        TTY$L_RB_ECHSTR                       !
026A   ;       +--------------------------------------------------------------+
026A   ;       !                         TTY$L_RB_PIC                         !
026A   ;       +--------------------------------------------------------------+
026A   ;       !                        TTY$L_RB_TERM                         !
026A   ;       +--------------------------------------------------------------+
026A   ;       !                        TTY$L_RB_MOD                          !
026A   ;       +--------------------------------------------------------------+
026A   ;       !                         TTY$L_RB_LIN                         !
026A   ;       +--------------------------------------------------------------+
026A   ;       !        TTY$W_RB_LINREST       !          TTY$W_RB_LINOFF     !
026A   ;       +--------------------------------------------------------------+
026A   ;       !        TTY$W_RB_TIMOS         !          TTY$W_RB_PRMLEN     !
026A   ;       +--------------------------------------------------------------+
026A   ;       !        TTY$W_RB_CPZORG        !          TTY$W_RB_CPZCUR     !
026A   ;       +--------------------------------------------------------------+
026A   ;       !        TTY$W_RB_PICLEN        !          TTY$W_RB_TXTOFF     !
026A   ;       +--------------------------------------------------------------+
026A   ;       !        TTY$W_RB_TXTECH        !          TTY$W_RB_TXTSIZ     !
026A   ;       +--------------------------------------------------------------+
026A   ;       !TTY$B_RB_RVFFIL!TTY$B_RB_RVFCLR!          TTY$W_RB_MODE       !
026A   ;       +--------------------------------------------------------------+
026A     ;          !                  TTY$A_RB_PRM or TTY$L_RB_DATA                      !
026A     ;          +-----------------------------------------------------------------+
026A     ;
026A     ;          4.  IRP$W_FUNC<0:6> ARE SET FOR A FAST CASE ON FUNCTION TYPE
026A     ;          5.  IRP$W_BOFF IS THE QUOTA FOR THE I/O
026A     ;          6.  IRP$W_BCNT IS THE READ REQUEST SIZE
026A     ;
026A     ; ITEM LIST TYPES AND PROCESSING:
026A     ;
026A     ;  TRM$_MODIFIERS:
026A     ;          32 BIT VALUE - SPECIFYES READ MODIFIERS CURRENT MODIFIERS ARE:
026A     ;                  TRM$M_TM_CVTLOW
026A     ;                  TRM$M_TM_DSABLMBX
026A     ;                  TRM$M_TM_NOECHO SETS TTY$V_ST_NOECHO
026A     ;                  TRM$M_TM_NOFILTR SETS TTY$V_ST_NOFILTR
026A     ;                  TRM$M_TM_PURGE
026A     ;                  TRM$M_TM_TIMED
026A     ;                  TRM$M_TM_TRMNOECHO
026A     ;                  TRM$M_TM_REFRESH SETS TTY$V_ST_REFRSH
026A     ;                  TRM$M_TM_NOEDIT
026A     ;                  TRM$M_TM_NORECALL
026A     ;              ALL OTHER'S MUST BE ZERO
026A     ;  TRM$_EDITMODE
026A     ;          VALUE TO SPECIFY THE TYPE OF READ TO BE ISSUED
026A     ;          CURRENT MODES ARE:
026A     ;                  TRM$K_EM_DEFAULT - NO SPECAIL FEATURES (DEFAULT)
026A     ;                  TRM$K_EM_EXTEDIT - EXTENDED EDITING CAPABILITIES
026A     ;                  TRM$K_EM_RDVERIFY - CHARACTER VALIDATING READ
026A     ;  TRM$_TIMEOUT
026A     ;          32 BIT VALUE SPECIFYING NUMBER OF SECONDS TO WAIT BETWEEN CHARACTERS.
026A     ;          SETS IO$M_TIMED IF SPECIFYED IN THE ITEM LIST.
026A     ;  TRM$_TERM
026A     ;          ADDRESS OF THE TERMINATOR BITMASK. A ZERO LENGTH INDICATES AN IMMEDIATE
026A     ;          SHORT FORM BITMASK.
026A     ;  TRM$_PROMPT
026A     ;          LENGTH AND ADDRESS OF PROMPT STRING. SETS TTY$V_ST_PROMPT AND
026A     ;          TTY$V_ST_EDITREAD.
026A     ;  TRM$_INISTRNG
026A     ;          LENGTH AND ADDRESS OF STRING TO LOAD INTO THE READ BUFFER AS IF
026A     ;          THE USER HAD TYPED IT.
026A     ;  TRM$_PICSTRNG
026A     ;          LENGTH AND ADDRESS OF THE PICTURE STRING (ONLY VALID WITH RDVERIFY
026A     ;          SET.
026A     ;  TRM$_FILLCHR
026A     ;          TWO BYTE VALUE OF THE FILL AND CLEAR CHARACTER FOR READ VERIFY
026A     ;  TRM$_INIOFFSET
026A     ;          16 BIT VALUE SPECIFYING HOWMANY CHARACTERS INTO THE INITIAL STRING
026A     ;          TO BEGIN ECHOING.  IF NON-ZERO THEN THE PROMPT STRING WILL NOT BE
026A     ;          ECHOED.  DEFAULT IS ZERO.
026A     ;
026A     ; INPUTS:
026A     ;
026A     ;          R3 = ADDRESS OF THE IRP FOR THIS REQUEST
026A     ;          R4 = CURRENT PCB
026A     ;          R5 = UCB ADDRESS
026A     ;          R6 = ASSIGNED CCB
026A     ;          R7 = FUNCTION CODE
```

```
                              026A   731 ;       AP = ADDRESS OF FIRST FUNCTION DEPENDENT QIO PARAMETER
                              026A   732 ;
                              026A   733 ; OUTPUTS:
                              026A   734 ;
                              026A   735 ;       THE I/O IF IN ERROR IS COMPLETED VIA EXE$ABORTIO
                              026A   736 ;       THE I/O IF VALID IS QUEUED TO THE DRIVER BY EXE$QIODRVPKT
                              026A   737 ;
                              026A   738 ; COMPLETION CODES:
                              026A   739 ;
                              026A   740 ;       SS$_ACCVIO - ACCESS VIOLATION ON BUFFER
                              026A   741 ;       SS$_EXQUOTA - OVER QUOTA FOR BUFFERED I/O
                              026A   742 ;       SS$_INSFMEM - INSUFFICIENT MEMORY
                              026A   743 ;       SS$_BADPARAM - ITEM LIST CONTAINED INVALID DATA.
                              026A   744 ;
                              026A   745 ;--
                              026A   746 TTY$FDTITEMREAD:
               0C AC     D5   026A   747         TSTL    P4(AP)                  ; P4 MUST BE ZERO
                  09     13   026D   748         BEQL    4$                      ; IT IS THEN CONTINUE
            50    14     3C   026F   749 2$:     MOVZWL  #SS$_BADPARAM,R0        ; SETUP R0 FOR ERROR
         00000000'GF     17   0272   750         JMP     G^EXE$ABORTIO           ; AND ABORT THE I/O
                              0278   751
                         B3   0278   752 4$:     BITW    #IO$M_CVTLOW!-          ; CHECK ALL FUNCTION MODIFIERS
                              0279   753                 IO$M_DSABLMBX!-
                              0279   754                 IO$M_NOECHO!-
                              0279   755                 IO$M_NOFILTR!-
                              0279   756                 IO$M_PURGE!-
                              0279   757                 IO$M_TIMED!-
                              0279   758                 IO$M_TRMNOECHO!-
                              0279   759                 IO$M_ESCAPE!-
                              0279   760                 IO$M_REFRESH,-
       20 A3    7FC0 8F           0279   761                 IRP$Q_FUNC(R3)
                  EF     12   027E   762         BNEQ    2$                      ; MAKE SURE ALL ARE OFF
                  57     D4   0280   763         CLRL    R7                      ; CLEAN OUT THE BUFFER LENGTH
                  58     D4   0282   764         CLRL    R8                      ; AND THE STATE BITS
         1A 48 A5  0C    E1   0284   765         BBC     #TT2$V_EDITING,UCB$L_DEVDEPND2(R5),3$; IS THIS AN EDITING READ
      58   00100000 8F   C8   0289   766         BISL    #TTY$M_ST_EDITING,R8    ; YES THEN SET EDITING
      58   00800000 8F   C8   0290   767         BISL    #TTY$M_ST_OVERSTRIKE,R8 ; IN OVERSTRIKE THEN SAY SO.
         07 48 A5  0D    E1   0297   768         BBC     #TT2$V_INSERT,UCB$L_DEVDEPND2(R5),3$; IF HE WANTS INSERT THEN
      58   00800000 8F   CA   029C   769         BICL    #TTY$M_ST_OVERSTRIKE,R8 ; CLEAR INSERT
                              02A3   770 ;
                              02A3   771 ; CHECK ACCESS TO READ BUFFER
                              02A3   772 ;
            50    6C     D0   02A3   773 3$:     MOVL    P1(AP),R0               ; GET BUFFER ADDRESS AND SIZE
         51    04 AC     3C   02A6   774         MOVZWL  P2(AP),R1               ;
                  05     12   02AA   775         BNEQ    5$                      ; IF NEQ THEN ACTUAL READ
                  08     E3   02AC   776         BBCS    #TTY$V_ST_EOL,-
            06 40 A3          02AE   777                 IRP$Q_TT_STATE(R3),10$  ; SET EOL AND BRANCH
         00000000'GF     16   02B1   778 5$:     JSB     G^EXE$READCHK           ; CHECK READ ACCESS FOR BUFFER
                              02B7   779                                         ; NO RETURN MEANS NO ACCESS
0C    20 A3    06    00  ED   02B7   780 10$:    CMPZV   #IRP$V_FCODE,#IRP$S_FCODE,IRP$W_FUNC(R3),#IO$_READPBLK; PASSALL?
                  10     13   02BD   781         BEQL    12$                     ; IF EQL THEN YES
            06    00     ED   02BF   782         CMPZV   #IRP$V_FCODE,#IRP$S_FCODE,-     ; TEMP READ PASSALL
         3A    20 A3          02C2   783                 IRP$W_FUNC(R3),#IO$_TTYREADALL
                  08     13   02C5   784         BEQL    12$                     ; NO, BRANCH
            06    00     ED   02C7   785         CMPZV   #IRP$V_FCODE,#IRP$S_FCODE,-     ; READ PASSALL W/PROMPT?
         3B    20 A3          02CA   786                 IRP$W_FUNC(R3),#IO$_TTYREADPALL
                  04     12   02CD   787         BNEQ    20$                     ; BRANCH IF NO
```

```
                                 02CF
                                 02CF     12$:
                                 02CF
                                 02CF              .IF DF CA$_MEASURE_IOT
                                 02CF
                                 02CF              BLBC    G^PMS$GL_DOSTATS,15$     ; IF FLAG OFF BYPASS NEXT INST.
                                 02CF              INCL    G^PMS$GL_PASSALL         ; ELSE INCR PASSALL COUNTER
                                 02CF
                                 02CF              .ENDC
                                 02CF
          00 58     02      E3   02CF     15$:     BBCS    #TTY$V_ST_PASALL,R8,20$ ; SET PASSALL MODE AND BR
                                 02D3     ;
                                 02D3              CHECK FOR ACCESS TO ITEM LIST.
                                 02D3     ;
             1000 8F        3C   02D3     20$:     MOVZWL  #TTY$M_ST_READ,-
               40 A3             02D7                      IRP$Q_TT_STATE(R3)          ; INIT AND ADD READ TO THE STATE BITS
     5E    00000058 8F      C2   02D9              SUBL    #TTY$K_IS_LENGTH,SP         ; ALLOCATE THE STACK STRUCTURE
               5B   5E      D0   02E0              MOVL    SP,R11                      ; MAKE R11 POINT TO THE STACK STRUCTURE
            08 AB   50      7D   02E3              MOVQ    R0,TTY$L_IS_BUF(R11)        ; SAVE THE USER'S BUFFER ADDRESS AND LENGTH
                                 02E7     ASSUME   TTY$L_IS_BUF+4 EQ TTY$L_IS_BUFLEN; STATE THE ASSUMPTION
                    38      BB   02E7              PUSHR   #^M<R3,R4,R5>               ; SAVE THE REGISTERS FOR EVERYONE
               5A   53      D0   02E9              MOVL    R3,R10                      ; KEEP THE IRP ADDRESS
 50     08 AC  02   00      EF   02EC              EXTZV   #0,#2,P3(AP),R0             ; GET THE ACCESS MODE
         00000000'GF        16   02F2              JSB     G^EXE$MAXACMODE             ; MAXIMIZE ACCESS MODE
               6B   50      D0   02F8              MOVL    R0,TTY$L_IS_ACMODE(R11)     ; KEEP IT IN THE STACK FOR LATER USE
               53   50      D0   02FB              MOVL    R0,R3                       ; GET THE ACCESS MODE INTO R3
           50   10 AC       D0   02FE              MOVL    P5(AP),R0                   ; GET THE ADDRESS OF THE ITEM LIST
           51   14 AC       3C   0302              MOVZWL  P6(AP),R1                   ; AND IT'S LENGTH
            1C AB   50      7D   0306              MOVQ    R0,TTY$L_IS_ITMLST(R11)     ; SAVE THE ADDRESS AND LENGTH
                                 030A     ASSUME   TTY$L_IS_ITMLST+4 EQ TTY$L_IS_LASTITM; STATE OUR ASSUMPTION
         00000000'GF        16   030A              JSB     G^EXE$PROBER                ; MAKE SURE WE CAN READ THE ITEM LIST
                03 50       E8   0310              BLBS    R0,30$                      ; YES THEN START TO PROCESS IT
                 02D0       31   0313              BRW     ITMREADERR                  ; ELSE RETURN HIM THE ERROR
                                 0316
                                 0316     ;
                                 0316     ; START TO PROCESS ITEM LIST
                                 0316     ;
       20 AB    1C AB       C0   0316     30$:     ADDL    TTY$L_IS_ITMLST(R11),TTY$L_IS_LASTITM(R11); MAKE THE LENGTH
                                 031B                                                  ; A POINTER TO THE END OF THE ITEMLIST
                                 031B     ;
                                 031B     ; CLEAN OUT THE STACK STRUCTURE
                                 031B     ;
                 30 AB      7C   031B              CLRQ    TTY$L_IS_PRM(R11)           ; ZERO THE PROMPT ADDRESS AND LENGTH
                 10 AB      7C   031E              CLRQ    TTY$L_IS_INI(R11)           ; THE INITIAL STRING ADDRESS AND LENGTH
                 40 AB      7C   0321              CLRQ    TTY$L_IS_TERM(R11)          ; TERMINATOR MASK ADR AND LENGTH
                 50 AB      D4   0324              CLRL    TTY$L_IS_TIMEOUT(R11)       ; THE TIMEOUT VALUE
                 24 AB      D4   0327              CLRL    TTY$L_IS_MODIFY(R11)        ; THE MODIFIER LONG WORD
                 28 AB      7C   032A              CLRQ    TTY$L_IS_PIC(R11)           ; THE PICTURE STRING ADDRESS AND LEGTH
                 54 AB      B4   032D              CLRW    TTY$W_IS_FILLCHR(R11)       ; FILL AND CLEAR CHARACTER
                 56 AB      B4   0330              CLRW    TTY$W_IS_INIOFF(R11)        ; AND THE INITIAL OFFSET
                 04 AB      D4   0333              CLRL    TTY$L_IS_EDITMODE(R11)      ; DEFAULT THE EDIT MODE
                 3C AB      D4   0336              CLRL    TTY$L_IS_SPECIFYED(R11)     ; ALSO THE ITEM SPECIFYED BIT MASK
                                 0339     ASSUME   TRM$_LASTITM LE 32                  ; MAKE SURE THAT THE ITEMS WILL
                                 0339                                                  ; FIT IN THE BITMASK.
                    07      E0   0339              BBS     #TT$V_LOWER,-               ; IS THIS LOWER CASE
              05 44 A5           033B                      UCB$L_DEVDEPEND(R5),ITEMLOOP; NO THEN CONTINUE ON
                    08      E2   033E              BBSS    #IO$V_CVTLOW,-
         00 24 AB       0340   845                      TTY$L_IS_MODIFY(R11),ITEMLOOP; YES THEN SET LOWER IN FUNCTION
                        0343   846
                        0343   847 ;
                        0343   848 ; MAIN ITEM LIST PROCESSING LOOP
                        0343   849 ;
                        0343   850 ITEMLOOP:
         59   1C AB  D0 0343   851 50$:     MOVL    TTY$L_IS_ITMLST(R11),R9 ; GET THE ADDRESS OF THIS ITEM
         1C AB   0C  C0 0347   852          ADDL    #TTY$K_IL_LENGTH,TTY$L_IS_ITMLST(R11); UPDATE THE COUNTER
   20 AB   1C AB     D1 034B   853          CMPL    TTY$L_IS_ITMLST(R11),TTY$L_IS_LASTITM(R11); ARE WE DONE
                 0F  15 0350   854          BLEQ    60$                           ; NO THEN GOTO THE DISPATCHER
                        0352   855
      20 AB      59  D1 0352   856          CMPL    R9,TTY$L_IS_LASTITM(R11); WAS THE ITEM LIST THE CORRECT LENGTH
                 03  12 0356   857          BNEQ    55$                           ; NO THEN BAD PARAMETER
               0033  31 0358   858          BRW     300$                          ; YES THEN ALLOCATE THE BUFFER AND BEGIN
               0281  31 035B   859 55$:     BRW     BDPRMERR                      ; GIVE ERROR
               0285  31 035E   860 57$:     BRW     ITMREADERR                    ; ERROR OUT
                        0361   861 ;
                        0361   862 ; ITEM LIST DISPATCHER
                        0361   863 ;
                        0361   864 ; INPUTS TO ROUTINES:
                        0361   865 ;
                        0361   866 ;        R4 = PCB ADDRESS
                        0361   867 ;        R5 = UCB ADDRESS
                        0361   868 ;        R7 = SIZE OF DATA AREA TO BE ADDED TO THE NORMAL READ DATA SECTION
                        0361   869 ;        R8 = SECOND LONG WORD OF THE UNIT STATE VECTOR
                        0361   870 ;        R9 = ADDRESS OF THIS ITEM BUFFER
                        0361   871 ;        R10 = IRP ADDRESS
                        0361   872 ;        R11 = STACK STRUCTURE ADDRESS
                        0361   873 ;
                        0361   874 ;        ALL OTHERS ARE SCRATCH
                        0361   875 ;
            08 A9    D5 0361   876 60$:     TSTL    TTY$L_IL_RETADR(R9)           ; MAKE SURE THE RETURN ADDRESS IS ZERO
                 F5  12 0364   877          BNEQ    55$                           ; ERROR IF NOT
      51   02 A9     3C 0366   878          MOVZWL  TTY$W_IL_TYPE(R9),R1          ; GET THE ITEM TYPE
         0A      51  D1 036A   879          CMPL    R1,#TRM$_LASTITM              ; IS THIS ITEM IN RANGE
                 EC  1E 036D   880          BGEQU   55$                           ; NO THEN ERROR
   E7 3C AB      51  E2 036F   881          BBSS    R1,TTY$L_IS_SPECIFYED(R11),55$; ONLY ALLOW ONE OF EACH ITEM
                        0374   882                                                ; TYPE.
                        0374   883          CASE    R1,TYPE=W,<- ;DISPATCH ON THE ITEM CODE
                        0374   884                  MODIFIERS,-
                        0374   885                  EDITMODE,-
                        0374   886                  TIMEOUT,-
                        0374   887                  TERM,-
                        0374   888                  PROMPT,-
                        0374   889                  INISTRNG,-
                        0374   890                  PICSTRNG,-
                        0374   891                  FILLCHR,-
                        0374   892                  INIOFFSET,-
                        0374   893                  ALTECHSTR>
                 CD  11 038C   894          BRB     55$
```

```
                                    038E     896 ;
                                    038E     897 ; ALLOCATE BUFFER AND DO FINAL VALIDATION
                                    038E     898 ;
                                    038E     899 300$:
                                    038E     900 ;
                                    038E     901 ; VALIDATION BEFORE ALLOCATING BUFFERS
                                    038E     902 ;
                    56 AB     B7    038E     903         DECW    TTY$W_IS_INIOFF(R11)     ; MAKE INDEX AN OFFSET
          14 AB     56 AB     B1    0391     904         CMPW    TTY$W_IS_INIOFF(R11),TTY$L_IS_INILEN(R11); DOES THE INITIAL
                                    0396     905                                          ; STRING OFFSET STAY WITHIN THE INITIAL STRING
                       C3     14    0396     906         BGTR    55$                      ; NO THEN ERROR OUT
                                    0398     907
                                    0398     908 ;
                                    0398     909 ; VALIDATION SPECIFIC TO READ VERIFY
                                    0398     910 ;
             04 AB     01     D1    0398     911         CMPL    #TRM$K_EM_RDVERIFY,TTY$L_IS_EDITMODE(R11); ARE WE A READ VERIFY
                       0F     12    039C     912         BNEQ    305$                     ; NO THEN DON'T DO THE VALIDATION
                    2C AB     D5    039E     913         TSTL    TTY$L_IS_PICLEN(R11)     ; ZERO LENGTH PICTURE STRING IS ILLEGAL
                       B8     13    03A1     914         BEQL    55$
          2C AB     14 AB     D1    03A3     915         CMPL    TTY$L_IS_INILEN(R11),TTY$L_IS_PICLEN(R11); IS THE PICTURE STRING
                       B1     12    03A8     916         BNEQ    55$                      ; LONG ENOUGH? NO THEN ERROR OUT
                    56 AB     B6    03AA     917         INCW    TTY$W_IS_INIOFF(R11)     ; UP THE INITIAL OFFSET
                                    03AD     918 305$:
          12 44 A5     06     E1    03AD     919         BBC     #TT$V_SCRIPT,UCB$L_DEVDEPEND(R5),310$; RTE TERMINAL LINE ?
                       57     D6    03B2     920         INCL    R7                       ; ADD RTE PROMPT SIZE
                    58 20     C8    03B4     921         BISL2   #TTY$M_ST_PROMPT,R8      ; ENSURE FUNCTION IS A RWP
              00000200 8F     C8    03B7     922         BISL2   #TTY$M_ST_EDITREAD,-
                    40 AA           03BD     923                 IRP$Q_TT_STATE(R10)      ; ENSURE FUNCTION IS A RWP
          00 2A AA     01     E2    03BF     924         BBSS    #IRP$V_FUNC,IRP$W_STS(R10),310$; ENSURE TRANFER DIRECTION RESET
                                    03C4     925 310$:
                                    03C4     926 ;
                                    03C4     927 ; ADD IN THE DATA REGION
                                    03C4     928 ; AND THE DATA BUFFER SIZE THEN ALLOCATE THE READ PACKET
                                    03C4     929 ;
      57      0000004A 8F     C0    03C4     930         ADDL    #TTY$L_RB_DATA,R7        ; ADD IN THE DATA REGION
                 57    0C AB  C0    03CB     931         ADDL    TTY$L_IS_BUFLEN(R11),R7  ; AND THE AREA FOR THE DATA
                    51 57     D0    03CF     932         MOVL    R7,R1                    ; SAVE THE LENGTH IN R1
             00000000'GF      16    03D2     933         JSB     G^EXE$BUFFRQUOTA         ; CHECK QUOTA
                    09 50     E9    03D8     934         BLBC    R0,315$                  ; ITEM READ ERROR? IF YES THEN HANDLE IT
             00000000'GF      16    03DB     935         JSB     G^EXE$ALLOCBUF           ; ALLOCATE THE BUFFER
                    03 50     E8    03E1     936         BLBS    R0,317$                  ; ERROR THEN HANDLE AS SUCH
                     01FF     31    03E4     937 315$:   BRW     ITMREADERR
                                    03E7     938 ;
                                    03E7     939 ; ADJUST BUFFERED I/O QUOTA
                                    03E7     940 ;
          50      0080 C4     D0    03E7     941 317$:   MOVL    PCB$L_JIB(R4),R0         ; GET JIB ADDRESS
                    51 51     3C    03EC     942         MOVZWL  R1,R1                    ; CONVERT TO LONG WORD COUNT
             20 A0     51     C2    03EF     943         SUBL    R1,JIB$L_BYTCNT(R0)      ; ADJUST QUOTA WORD
                                    03F3     944 ;
                                    03F3     945 ; SETUP REGISTERS AND ADDITIONAL CONSTANT FIELDS
                                    03F3     946 ;
                    57 52     D0    03F3     947         MOVL    R2,R7                    ; GET THE READ BUFER ADDRESS
                 2C AA 57     D0    03F6     948         MOVL    R7,IRP$L_SVAPTE(R10)     ; SAVE THE ADDRESS
                 30 AA 51     B0    03FA     949         MOVW    R1,IRP$W_BOFF(R10)       ; SAVE THE SIZE AS A QUOTA
                 08 A7 51     B0    03FE     950         MOVW    R1,TTY$W_RB_SIZE(R7)     ; SAVE THE SIZE FOR THE SYSTEM
             59    4A A7      9E    0402     951         MOVAB   TTY$L_RB_DATA(R7),R9     ; AND GET THE BEGINNING OF THE DATA AREA
                                    0406     952 ;
```

```
                        0406  ; FILL IN NEWLY ALLOCATED READ BUFFER
                        0406  ;
                        0406  ; REGISTERS ARE AS FOLLOWS:
                        0406  ;
                        0406  ;          R4 = PCB ADDRESS
                        0406  ;          R5 = UCB ADDRESS
                        0406  ;          R7 = READ BUFFER ADDRESS
                        0406  ;          R9 = BEGINNING OF THE DATA AREA (USED AS POINTER TO THE NEXT FREE
                        0406  ;                 ADDRESSES IN THE READ BUFFER)
                        0406  ;          R10 = IRP ADDRESS
                        0406  ;          R11 = STACK BUFFER ADDRESS
                        0406  ;
                        0406  ;
                        0406  ; COPY IN PROMPT AND ASSOCIATED DATA
                        0406  ;
      50    30 AB   7D  0406           MOVQ    TTY$L_IS_PRM(R11),R0     ; GET THE ADDRESS AND LENGTH OF THE
      34 A7    51   B0  040A           MOVW    R1,TTY$W_RB_PRMLEN(R7)   ; COPY IN THE LENGTH
      5B 58    05   E1  040E           BBC     #TTY$V_ST_PROMPT,R8,340$; DO WE HAVE A PROMPT
                        0412  ;
                        0412  ; HANDLE RTE MODE
                        0412  ;
   0D 44 A5    06   E1  0412           BBC     #TT$V_SCRIPT,UCB$L_DEVDEPEND(R5),320$; IS THIS RTE MODE
6941   00000000'9F  90  0417           MOVB    @#PMS$GB_PROMPT,(R9)[R1]; PUT THE CHARACTER IN PLACE
            34 A7   B6  041F           INCW    TTY$W_RB_PRMLEN(R7)      ; INCREMENT THE PROMPT LENGTH
               59   D6  0422           INCL    R9                       ; ADD 1 TO THE CURRENT ADDRESS
               51   D5  0424  320$:    TSTL    R1                       ; ANY CHARACTERS TO MOVE
               45   13  0426           BEQL    340$                     ; NO THEN GO AROUND
         59    51   C0  0428           ADDL    R1,R9                    ; UPDATE THE ENDING ADDRESS
                        042B  ;
                        042B  ; TRANSLATE FOR NON-LOWER CASE TERMINALS IF NECESSARY
                        042B  ;
   13 48 A5    0E   E0  042B           BBS     #TT2$V_FALLBACK,UCB$L_DEVDEPND2(R5),327$
               07   E0  0430           BBS     #TT$V_LOWER,-            ; IS THIS LOWER CASE
         33 44 A5       0432                   UCB$L_DEVDEPEND(R5),330$; NO THEN CONTINUE ON
      2F 58    02   E0  0435  325$:    BBS     #TTY$V_ST_PASALL,R8,330$         ; NO TRANSLATE IF PASSALL
               00   E0  0439           BBS     #TT$V_PASSALL,-
         2A 44 A5       043B                   UCB$L_DEVDEPEND(R5),330$; CHECK FOR PASSALL OR PASTHRU
               12   E0  043E           BBS     #TT2$V_PASTHRU,-
         25 48 A5       0440                   UCB$L_DEVDEPND2(R5),330$; ...
                        0443  327$:
         0180 8F    BB  0443           PUSHR   #^M<R7,R8>                       ; NO - THEN SAVE R7 AND R8
               58   D4  0447           CLRL    R8                       ; ASSUME NO LOWER CASE
               07   E0  0449           BBS     #TT$V_LOWER,-            ; IS THIS LOWER CASE
         03 44 A5       044B                   UCB$L_DEVDEPEND(R5),329$; NO THEN CONTINUE ON
         58    01   D0  044E           MOVL    #1,R8                    ; LOW BIT SET MEANS TRANSLATE TO LOWER
      52    4A A7   9E  0451  329$:    MOVAB   TTY$A_RB_PRM(R7),R2      ; WE MUST MOVE TRANSLATED SO GET DEST ADDRES
      57    38 AB   D0  0455           MOVL    TTY$L_IS_PRMBUF(R11),R7  ; GET THE USERS BUFFER LENGTH (FOR FALLBACK
         56    50   D0  0459           MOVL    R0,R6                    ; GET THE ADDRESS OF THE STRING
         53    51   D0  045C           MOVL    R1,R3                    ; AND LENGTH TO MOVE.
            0841    30  045F           BSBW    MOVE_TRANSLATE           ; THEN TRANSLATE UNTILL THE HART'S CONTENT
         0180 8F    BA  0462           POPR    #^M<R7,R8>               ; RESTORE R7
               05   11  0466           BRB     340$
                        0468
                        0468
4A A7    60    51   28  0468  330$:    MOVC3   R1,(R0),TTY$A_RB_PRM(R7)         ; MOVE IN THE CHARACTERS IN THE PROM
                        046D  340$:
                        046D  ;
```

```
                       046D  1010 ; setup the data field
                       046D  1011 ;
        67    59   D0  046D  1012          MOVL    R9,TTY$L_RB_TXT(R7)      ; SAVE THE ADDRESS OF THE BEGINNING
     2C A7    59   D0  0470  1013          MOVL    R9,TTY$L_RB_LIN(R7)      ; THE DATA IN TXT AND LIN
        30 A7      B4  0474  1014          CLRW    TTY$W_RB_LINOFF(R7)      ; ZERO OUT FIELDS IN THE READ BUFFER
        32 A7      B4  0477  1015          CLRW    TTY$W_RB_LINREST(R7)
        3C A7      B4  047A  1016          CLRW    TTY$W_RB_TXTOFF(R7)
        44 A7      B4  047D  1017          CLRW    TTY$W_RB_MODE(R7)        ; CLEAN OUT THE REPROMPT MODES
        2A A7      B4  0480  1018          CLRW    TTY$W_RB_RDSTATE(R7)
        14 A7      D4  0483  1019          CLRL    TTY$L_RB_ECHSTR(R7)      ; CLEAN THE ECHO STRING
        59  0C AB  C0  0486  1020          ADDL    TTY$L_IS_BUFLEN(R11),R9  ; ADD IN THE BUFFER TO OUR COUNTS
     40 A7  0C AB  B0  048A  1021          MOVW    TTY$L_IS_BUFLEN(R11),TTY$W_RB_TXTSIZ(R7); MOVE IN THE BUFFER LENGTH
     04 A7  08 AB  D0  048F  1022          MOVL    TTY$L_IS_BUF(R11),TTY$L_RB_UVA(R7); AND SAVE THE USERS ADDRESS
                       0494  1023 ;
                       0494  1024 ; MOVE IN INITIAL STRING
                       0494  1025 ;
  53 3C AB     05  E1  0494  1026          BBC     #TRM$_INISTRNG,TTY$L_IS_SPECIFYED(R11),350$; DO WE HAVE A STRING
     50    10 AB   7D  0499  1027          MOVQ    TTY$L_IS_INI(R11),R0     ; GET THE ADDRESS AND LENGTH
     3C A7     51  B0  049D  1028          MOVW    R1,TTY$W_RB_TXTOFF(R7)   ; PUT THE LENGTH IN THE RIGHT PLACE
     30 A7     51  B0  04A1  1029          MOVW    R1,TTY$W_RB_LINOFF(R7)   ; AND THE LINE COUNTER
           45      13  04A5  1030          BEQL    350$                     ; NO INITIAL STRING THEN NO NEED TO
                       04A7  1031                                           ; CHECK FOR UPPER CASE
                       04A7  1032 ;
                       04A7  1033 ; TRANSLATE INITIAL STRING FROM LOWER CASE TO UPPER CASE IF NECESSARY
                       04A7  1034 ;
     55    08 AE   D0  04A7  1035          MOVL    8(SP),R5                 ; RESTORE THE UCB ADDRESS
  13 48 A5     0E  E0  04AB  1036          BBS     #TT2$V_FALLBACK,UCB$L_DEVDEPND2(R5),344$; FALLBACK THEN TRANSLATE
  32 24 AB     08  E1  04B0  1037          BBC     #TRM$V_TM_CVTLOW,TTY$L_IS_MODIFY(R11),345$; ARE WE CONVERTING
     2E 58     02  E0  04B5  1038 342$:    BBS     #TTY$V_ST_PASALL,R8,345$          ; NO TRANSLATE IF PASSALL
               00  E0  04B9  1039          BBS     #TT$V_PASSALL,-
     29 44 A5          04BB  1040                  UCB$L_DEVDEPEND(R5),345$          ; CHECK FOR PASSALL OR PASTHRU
               12  E0  04BE  1041          BBS     #TT2$V_PASTHRU,-
     24 48 A5          04C0  1042                  UCB$L_DEVDEPND2(R5),345$          ; ...
                       04C3  1043 344$:
     0180 8F       BB  04C3  1044          PUSHR   #^M<R7,R8>                        ; NO - THEN SAVE R7 AND R8
           58      D4  04C7  1045          CLRL    R8                       ; ASSUME NO LOWER CASE
           08      E1  04C9  1046          BBC     #TRM$V_TM_CVTLOW,-
     03 24 AB          04CB  1047                  TTY$L_IS_MODIFY(R11),347$; BRANCH IF UPPER CASE ONLY
     58    01      D0  04CE  1048          MOVL    #1,R8                    ; LOW BIT SET MEANS TRANSLATE TO LOWER
     52    67      D0  04D1  1049 347$:    MOVL    TTY$L_RB_TXT(R7),R2      ; WE MUST MOVE TRANSLATED SO GET DEST ADDRES
  57    18 AB      D0  04D4  1050          MOVL    TTY$L_IS_INIBUF(R11),R7  ; GET THE USERS BUFFER LENGTH
     56    50      D0  04D8  1051          MOVL    R0,R6                    ; GET THE ADDRESS OF THE STRING
     53    51      D0  04DB  1052          MOVL    R1,R3                    ; AND LENGTH TO MOVE.
         07C2      30  04DE  1053          BSBW    MOVE_TRANSLATE           ; THEN TRANSLATE UNTILL THE HART'S CONTENT
     0180 8F       BA  04E1  1054          POPR    #^M<R7,R8>               ; RESTORE R7
           05      11  04E5  1055          BRB     350$
 00 B7    60   51  28  04E7  1056 345$:    MOVC3   R1,(R0),@TTY$L_RB_TXT(R7)         ; MOVE IN THE INITIAL STRING
                       04EC  1057 350$:
                       04EC  1058 ;
                       04EC  1059 ; HANDLE INITIAL STRING OFFSET
                       04EC  1060 ;
  26 3C AB     08  E1  04EC  1061          BBC     #TRM$_INIOFFSET,TTY$L_IS_SPECIFYED(R11),360$; NOT SPECIFYED
                       04F1  1062 ;
                       04F1  1063 ; CHECK FOR READ VERIFY
                       04F1  1064 ;
     04 AB     01  D1  04F1  1065          CMPL    #TRM$K_EM_RDVERIFY,TTY$L_IS_EDITMODE(R11); ARE WE READ VERIFYING
               07  12  04F5  1066          BNEQ    355$                     ; NO THEN HANDLE NORMALY
```

```
            30 A7    56 AB   D0  04F7  1067          MOVL    TTY$W_IS_INIOFF(R11),TTY$W_RB_LINOFF(R7); SAVE THE INITIAL OFFSET
                        19   11  04FC  1068          BRB     360$
                                 04FE  1069 ;
                                 04FE  1070 ; NORMAL INITIAL OFFSETS
                                 04FE  1071 ;
                     56 AB   B5  04FE  1072 355$:    TSTW    TTY$W_IS_INIOFF(R11)      ; IS THERE REALY AN INITIAL OFFSET?
                        14   19  0501  1073          BLSS    360$                      ; no then do nothing
                     56 AB   83  0503  1074          SUBB3   TTY$W_IS_INIOFF(R11),-    ; GET THE LENGTH TO ECHO
            0B A7    14 AB       0506  1075                  TTY$L_IS_INILEN(R11),TTY$B_RB_ECHLEN(R7)
               51    56 AB   3C  050A  1076          MOVZWL  TTY$W_IS_INIOFF(R11),R1   ; GET THE INITIAL OFFSET INTO A LONG WORD
                  67    51   C1  050E  1077          ADDL3   R1,TTY$L_RB_TXT(R7),-     ; GET THE LOCATION OF THE FIRST CHARACTER
                     14 A7       0511  1078                  TTY$L_RB_ECHSTR(R7)
            44 A7       02   9B  0513  1079          MOVZBW  #TTY$K_ER_ECHLINE,TTY$W_RB_MODE(R7); AND SETUP THE ECHOING CORRECTLY
                                 0517  1080 ;
                                 0517  1081 ; MOVE IN PICTURE STRING
                                 0517  1082 ;
         14 3C AB       06   E1  0517  1083 360$:    BBC     #TRM$_PICSTRNG,TTY$L_IS_SPECIFYED(R11),370$
            50       28 AB   7D  051C  1084          MOVQ    TTY$L_IS_PIC(R11),R0      ; GET THE ADDRESS AND LENGTH
            3E A7       51   B0  0520  1085          MOVW    R1,TTY$W_RB_PICLEN(R7)    ; KEEP THE PICTURE STRING LENGTH
            18 A7       59   D0  0524  1086          MOVL    R9,TTY$L_RB_PIC(R7)       ; AND THE ADDRESS OF IT IN THE BUFFER
                  59    51   C0  0528  1087          ADDL    R1,R9                     ; UPDATE THE POINTER
      18 B7       60    51   28  052B  1088          MOVC3   R1,(R0),@TTY$L_RB_PIC(R7)         ; MOVE IN THE PICTURE STRING
                                 0530  1089 370$:
                                 0530  1090 ;
                                 0530  1091 ; MOVE TERMINATOR MASK INTO BUFFER
                                 0530  1092 ;
   1C A7    00000000'EF      9E  0530  1093          MOVAB   TTY$A_STANDARD,TTY$L_RB_TERM(R7); SETUP DEFAULT TERMINATOR MASK
         11 3C AB       03   E1  0538  1094          BBC     #TRM$_TERM,TTY$L_IS_SPECIFYED(R11),390$; DID THE USER SPECIFY
            50       40 AB   7D  053D  1095          MOVQ    TTY$L_IS_TERM(R11),R0     ; A TERMINATOR MASK, YES THEN GET IT
            1C A7       59   D0  0541  1096          MOVL    R9,TTY$L_RB_TERM(R7)      ; TELL WHERE THE MASK WILL END UP
69    20    00    60    51   2C  0545  1097          MOVC5   R1,(R0),#0,#32,(R9)       ; MOVE THE MASK IN ZEROING UNUSED BITS
                  59    53   D0  054B  1098          MOVL    R3,R9                     ; MOVE R9 TO THE END OF USED SPACE
                                 054E  1099 390$:
         1C 3C AB       09   E1  054E  1100          BBC     #TRM$_ALTECHSTR,TTY$L_IS_SPECIFYED(R11),400$
            50       48 AB   7D  0553  1101          MOVQ    TTY$L_IS_AES(R11),R0      ; GET THE ALTERNATE ECHO LEN AND ADR
            28 A7       51   B0  0557  1102          MOVW    R1,TTY$W_RB_AESLEN(R7)    ; MOVE IN THE LENGTH
                        12   13  055B  1103          BEQL    400$                      ; NO LENGTH THEN DON'T BOTHER
            24 A7       59   D0  055D  1104          MOVL    R9,TTY$L_RB_AES(R7)       ; ELSE GET THE ADDRESS
               69 60    51   28  0561  1105          MOVC3   R1,(R0),(R9)              ; AND MOVE IN THE DATA
                  59    53   D0  0565  1106          MOVL    R3,R9                     ; UPDATE THE END ADDRESS
      58    02000000 8F      C8  0568  1107          BISL    #TTY$M_ST_ECHAES,R8       ; SET THE FLAG INDICATING FIRST CHARACTER
                                 056F  1108 400$:
                                 056F  1109 ;
                                 056F  1110 ; SETUP TIMEOUT
                                 056F  1111 ;
            36 A7    50 AB   B0  056F  1112          MOVW    TTY$L_IS_TIMEOUT(R11),TTY$W_RB_TIMOS(R7); MOVE TIMEOUT IN
                                 0574  1113 ;
                                 0574  1114 ; MOVE FILL CHARACTERS INTO THEIR PLACE
                                 0574  1115 ;
            46 A7    54 AB   B0  0574  1116          MOVW    TTY$W_IS_FILLCHR(R11),TTY$B_RB_RVFCLR(R7); MOVE THE FILL
                                 0579  1117                                            ; AND CLEAR CHARACTER INTO PLACE
                                 0579  1118 ASSUME TTY$B_RB_RVFCLR+1 EQ TTY$B_RB_RVFFIL
                                 0579  1119 ;
                                 0579  1120 ; IF THIS IS A READ VERIFY THEN WE HAVE TO SKIP MARKERS
                                 0579  1121 ;
            04 AB       01   D1  0579  1122          CMPL    #TRM$K_EM_RDVERIFY,TTY$L_IS_EDITMODE(R11); MAKE SURE WE ARE IN READ
                        3C   12  057D  1123          BNEQ    650$                      ; NO THEN SKIP THE WHOLE THING
```

```
          50   30 A7   3C  057F  1124         MOVZWL  TTY$W_RB_LINOFF(R7),R0  ; GET THE INITIAL OFFSET
               30 A7   B7  0583  1125         DECW    TTY$W_RB_LINOFF(R7)     ; MOVE BACK ONE CHARACTER TO MAKE OFFSET
                           0586  1126                                         ; AN INDEX ONLY AFFECTIVE FOR RIGHT FIELDS
       18 24 AB    11  E0  0586  1127         BBS     #TRM$V_TM_R_JUST,TTY$L_IS_MODIFY(R11),640$; LITTLE WORK FOR RIGHT JU
            18 B740    95  058B  1128 600$:   TSTB    @TTY$L_RB_PIC(R7)[R0]   ; IS THIS A MARKER CHARACTER
                   0E  12  058F  1129         BNEQ    610$                    ; NO THEN WE ARE DONE
                   50  D6  0591  1130         INCL    R0                      ; ELSE MOVE OVER 1 CHARACTER
       3C A7       50  B1  0593  1131         CMPW    R0,TTY$W_RB_TXTOFF(R7)  ; ARE WE AT THE END OF THE FIELD
                   F2  19  0597  1132         BLSS    600$                    ; NO THEN CONTINUE
          50   30 A7   3C  0599  1133         MOVZWL  TTY$W_RB_LINOFF(R7),R0  ; GET THE INITIAL OFFSET
                   50  D6  059D  1134         INCL    R0                      ; ACCOUNT FOR THE RIGHT JUSTIFY CHANGE
       30 A7       50  B0  059F  1135 610$:   MOVW    R0,TTY$W_RB_LINOFF(R7)  ; UPDATE THE OFFSET
       02 58       03  E1  05A3  1136 640$:   BBC     #TTY$V_ST_NOECHO,R8,645$; IS THIS A NOECHO FIELD
                   50  D4  05A7  1137         CLRL    R0                      ; YES THEN DON'T ECHO ANY INITIAL STRING
       14 A7   4A A7   9E  05A9  1138 645$:   MOVAB   TTY$A_RB_PRM(R7),TTY$L_RB_ECHSTR(R7); GET THE ADDRESS TO START
 0B A7    50   34 A7   81  05AE  1139         ADDB3   TTY$W_RB_PRMLEN(R7),R0,TTY$B_RB_ECHLEN(R7); AND THE LENGTH TO ECHO
               32 A7   B4  05B4  1140         CLRW    TTY$W_RB_LINREST(R7)    ; INIT FIELD
       44 A7       0A  9B  05B7  1141         MOVZBW  #TTY$K_ER_RVECHO,TTY$W_RB_MODE(R7); SETUP THE ECHO MODE
                           05BB  1142 650$:
                           05BB  1143 ;
                           05BB  1144 ; CLEANUP AND QUEUE THE PACKET
                           05BB  1145 ;
                   38  BA  05BB  1146         POPR    #^M<R3,R4,R5>           ; restore the registers
       20 A7   24 AB   D0  05BD  1147         MOVL    TTY$L_IS_MODIFY(R11),TTY$L_RB_MOD(R7); MOVE THE MODIFIERS
                           05C2  1148                                         ; INTO PLACE
                           05C2  1149
                           05C2  1150 ;
                           05C2  1151 ; TEMPORARYLY SETUP THE MODIFIER BITS IN THE FUNCTION CODE
                           05C2  1152 ;
                           05C2  1153
                       CA  05C2  1154         BICL    #^C<TRM$M_TM_CVTLOW!-   ; CLEAR THE NEW MODIFIER BITS
                           05C3  1155                 TRM$M_TM_DSABLMBX!-
                           05C3  1156                 TRM$M_TM_NOECHO!-
                           05C3  1157                 TRM$M_TM_NOFILTR!-
                           05C3  1158                 TRM$M_TM_PURGE!-
                           05C3  1159                 TRM$M_TM_TIMED!-
                           05C3  1160                 TRM$M_TM_TRMNOECHO!-
                           05C3  1161                 TRM$M_TM_ESCAPE!-
                           05C3  1162                 TRM$M_TM_REFRESH>,-
 24 AB    FFFF803F 8F              05C3  1163                 TTY$L_IS_MODIFY(R11)
                           05CA  1164
               24 AB   A8  05CA  1165         BISW    TTY$L_IS_MODIFY(R11),-  ; GET THE MODIFIERS OUT OF OUR WORD
               20 A3       05CD  1166                 IRP$W_FUNC(R3)          ; AND PUT THEM IN THE FUNCTION CODE SLOT
                           05CF  1167 ; END TEMPORARY
       44 A3       58  D0  05CF  1168         MOVL    R8,IRP$Q_TT_STATE+4(R3) ; SETUP STATE QUAD WORD
 20 A3    06   00  00  F0  05D3  1169         INSV    #TTY$C_FC_READ,#IRP$V_FCODE,#IRP$S_FCODE,IRP$W_FUNC(R3)
          00000000'GF  17  05D9  1170         JMP     G^EXE$QIODRVPKT
```

```
                         05DF  ;
                         05DF  ; ERROR HANDLEING ROUTINES
                         05DF  ;
                         05DF  BDPRMERR:
     51    02 A9    3C   05DF          MOVZWL  TTY$W_IL_TYPE(R9),R1     ; RETURN THE BAD ITEM # IN R1
        50    14    3C   05E3          MOVZWL  #SS$_BADPARAM,R0         ; AND THE ERROR IN R0
                         05E6  ITMREADERR:
              38    BA   05E6          POPR    #^M<R3,R4,R5>            ; RESTORE STATE
  00000000'GF       17   05E8          JMP     G^EXE$ABORTIO            ; and abort the IO
```

```
                          05EE   1182
                          05EE   1183 ;++
                          05EE   1184 ; ALTECHSTR
                          05EE   1185 ;
                          05EE   1186 ; DESCRIPTION:
                          05EE   1187 ;
                          05EE   1188 ;         STRING TO BE ECHOED UPON THE FIRST VALID CHARACTER
                          05EE   1189 ; THAT IS TYPED.
                          05EE   1190 ;--
                          05EE   1191 ALTECHSTR:
 01     04 AB      D1     05EE   1192          CMPL    TTY$L_IS_EDITMODE(R11),#TRM$K_EM_RDVERIFY; ARE WE A READ VERIFY READ
           EB      12     05F2   1193          BNEQ    BDPRMERR                   ; NO THEN ERROR OUT
 50     04 A9      D0     05F4   1194          MOVL    TTY$L_IL_ADR(R9),R0        ; GET THE ADDRESS OF THE PROMPT
     51    69      3C     05F8   1195          MOVZWL  TTY$W_IL_LEN(R9),R1        ; AND THE LENGTH
           17      13     05FB   1196          BEQL    20$                        ; NO PROMPT THEN DO NOTHING
 48 AB     50      7D     05FD   1197          MOVQ    R0,TTY$L_IS_AES(R11)       ; SAVE THE PROMPT LENGTH AND ADDRESS
                          0601   1198 ASSUME   TTY$L_IS_AES+4 EQ TTY$L_IS_AESLEN
     53    6B      D0     0601   1199          MOVL    TTY$L_IS_ACMODE(R11),R3    ; SETUP OUR ACCESS MODE
00000000'GF        16     0604   1200          JSB     G^EXE$PROBER               ; CHECK THE ACCESS ON THE BUFFER
        03 50      E8     060A   1201          BLBS    R0,10$                     ; NO ERROR THEN CONTINUE
         FFD6      31     060D   1202          BRW     ITMREADERR                 ; ELSE ERROR OUT
 57     4C AB      C0     0610   1203 10$:     ADDL    TTY$L_IS_AESLEN(R11),R7    ; ADD IN THE LENGTH
         FD2C      31     0614   1204 20$:     BRW     ITEMLOOP                   ; CONTINUE ON NORMALY
                          0617   1205
                          0617   1206
```

```
                                 0617
                                 0617  ;++
                                 0617  ; EDITMODE
                                 0617  ;
                                 0617  ; DESCRIPTION
                                 0617  ;        VALIDATES ARGUMENTS AND SETS STATE BITS FOR THE DIFFERENT TYPE OF
                                 0617  ; EDIT MODES.
                                 0617  ;--
                                 0617  EDITMODE:
                      69    B5   0617          TSTW    TTY$W IL_LEN(R9)          ; LENGTH MUST BE ZERO
                      C4    12   0619          BNEQ    BDPRMERR
                   08 A9    D5   061B          TSTL    TTY$L IL_RETADR(R9)       ; ALSO THE SECOND ADDRESS
                      BF    12   061E          BNEQ    BDPRMERR
             50    04 A9    D0   0620          MOVL    TTY$L IL_ADR(R9),R0       ; GET THE MODE
                50    01    D1   0624          CMPL    #TRM$K EM_RDVERIFY,R0     ; AND CHECK IT FOR VALITITY
                      B6    14   0627          BGTR    BDPRMERR                  ; NOT VALID THEN ERROR
                      0F    12   0629          BNEQ    10$                       ; NOT READ VERIFY THEN SKIP
  40 AA    00000400 8F    C8   062B          BISL    #TTY$M_ST_RDVERIFY,IRP$Q_TT_STATE(R10); SETTING READVERIFY STATE
     58    00100000 8F    C8   0633          BISL    #TTY$M_ST_EDITING,R8      ; YES THEN SET EDITING
        04 AB       50    D0   063A  10$:    MOVL    R0,TTY$L_IS_EDITMODE(R11); SAVE THE MODE
                                 063E
                    FD02    31   063E          BRW     ITEMLOOP
```

```
                          0641   1231
                          0641   1232 ;++
                          0641   1233 ;FILLCHR
                          0641   1234 ;
                          0641   1235 ; DESCRIPTION:
                          0641   1236 ;         MOVE THE FILL CHARACTERS INTO THE STACK STRUCTURE.
                          0641   1237 ; FILL CHARACTERS ARE USED BY READ VERIFY.
                          0641   1238 ;--
                          0641   1239 FILLCHR:
         01    04 AB  D1  0641   1240          CMPL    TTY$L_IS_EDITMODE(R11),#TRM$K_EM_RDVERIFY; ARE WE A READ VERIFY READ
                  71  12  0645   1241          BNEQ    GOBAD                    ; NO THEN ERROR OUT
                  69  B5  0647   1242          TSTW    TTY$W_IL_LEN(R9)         ; LENGTH MUST BE ZERO
                  6D  12  0649   1243          BNEQ    GOBAD
               08 A9  D5  064B   1244          TSTL    TTY$L_IL_RETADR(R9)      ; ALSO THE SECOND ADDRESS
                  68  12  064E   1245          BNEQ    GOBAD
      54 AB    04 A9  B0  0650   1246          MOVW    TTY$L_IL_ADR(R9),TTY$W_IS_FILLCHR(R11); MMOVE IN THE FILL CHARACTERS
                FCEB  31  0655   1247          BRW     ITEMLOOP                 ; THEN CONTINUE ON WITH THE LIST
```

```
                        0658  1249 ;++
                        0658  1250 ; INIOFFSET
                        0658  1251 ;
                        0658  1252 ; DESCRIPTION:
                        0658  1253 ;         SAVE THE OFFSET INTO THE INITIAL STRING, RANGE CHECKING MUST BE
                        0658  1254 ; DONE LATER TO ALLOW THE ITEMS TO APPEAR IN ANY ORDER.
                        0658  1255 INIOFFSET:
                 69  B5 0658  1256         TSTW    TTY$W_IL_LEN(R9)          ; LENGTH MUST BE ZERO
                 5C  12 065A  1257         BNEQ    GOBAD
              08 A9  D5 065C  1258         TSTL    TTY$L_IL_RETADR(R9)       ; ALSO THE SECOND ADDRESS
                 57  12 065F  1259         BNEQ    GOBAD
      56 AB   04 A9  B0 0661  1260         MOVW    TTY$L_IL_ADR(R9),TTY$W_IS_INIOFF(R11); MOVE IN THE OFFSET
                 50  19 0666  1261         BLSS    GOBAD                     ; NO NEGITIVE NUMBERS ALLOWED
               FCD8  31 0668  1262         BRW     ITEMLOOP                  ; THEN CONTINUE ON WITH THE LIST
```

```
                            066B  ;++
                            066B  ; INISTRNG
                            066B  ;
                            066B  ; DESCRIPTION:
                            066B  ;          CHECK ACCESS TO THE USERS INITIAL STRING.
                            066B  ;--
                            066B  INISTRNG:
         50    04 A9    D0  066B            MOVL     TTY$L_IL_ADR(R9),R0         ; GET THE ADDRESS OF THE INITIAL STRING
             51    69    3C  066F            MOVZWL   TTY$W_IL_LEN(R9),R1         ; AND THE LENGTH
         10 AB    50    7D  0672            MOVQ     R0,TTY$L_IS_INI(R11)        ; SAVE THE INITIAL STRING LENGTH AND ADDRESS
                            0676  ASSUME    TTY$L_IS_INI+4 EQ TTY$L_IS_INILEN
         18 AB    51    D0  0676            MOVL     R1,TTY$L_IS_INIBUF(R11)     ; KEEP THE LENGTH
                  20    13  067A            BEQL     20$                         ; NO INITIAL STRING THEN DO NOTHING
            53    6B    D0  067C            MOVL     TTY$L_IS_ACMODE(R11),R3     ; SETUP OUR ACCESS MODE
      00000000'GF    16  067F            JSB      G^EXE$PROBER                ; CHECK THE ACCESS ON THE BUFFER
              17 50    E9  0685            BLBC     R0,10$                      ; NO ERROR THEN CONTINUE
      15 48 A5    0E    E0  0688            BBS      #TT2$V_FALLBACK,UCB$L_DEVDEPND2(R5),30$
      0C AB    14 AB    D1  068D  15$:      CMPL     TTY$L_IS_INILEN(R11),TTY$L_IS_BUFLEN(R11); DOES THIS STRING FIT INTO
                  24    14  0692            BGTR     GOBAD                       ; NO THEN INFORM THE USER.
40 AA    00000200 8F    C8  0694            BISL     #TTY$M_ST_EDITREAD,IRP$Q_TT_STATE(R10); MAKE SURE THE PROMPT IS ECHO
                FCA4    31  069C  20$:      BRW      ITEMLOOP                    ; CONTINUE ON NORMALY
                FF44    31  069F  10$:      BRW      ITMREADERR                  ; ELSE ERROR OUT
                  59    DD  06A2  30$:      PUSHL    R9                          ; SAVE R9 OVER THE CALL
         51    10 AB    D0  06A4            MOVL     TTY$L_IS_INI(R11),R1        ; RESTORE ADDRESS AND LENGTH
         50    14 AB    D0  06A8            MOVL     TTY$L_IS_INILEN(R11),R0     ; AND THE LENGTH
                06B7    30  06AC            BSBW     ADDFALL                     ; CALCULATE THE ADDITIONAL
                            06AF                                                 ; CHARACER COUNT OF FALLBACK
         14 AB    59    C0  06AF            ADDL     R9,TTY$L_IS_INILEN(R11)     ;ADD IN THE COUNT
                  59  8ED0  06B3            POPL     R9                          ; RESTORE R9
                  D5    11  06B6            BRB      15$                         ;
                            06B8
                FF24    31  06B8  GOBAD:    BRW      BDPRMERR
```

```
                               06BB  1297 ;++
                               06BB  1298 ; MODIFIERS
                               06BB  1299 ;
                               06BB  1300 ; DESCRIPTION
                               06BB  1301 ;        PROCESS MODIFIER ITEM LIST ENTRY.  VALIDATES ARGUMENTS AND
                               06BB  1302 ; ABORTS ON ERRORS.  SETS APPROPRIATE BITS IN THE UNIT STATE VECTOR
                               06BB  1303 ;--
                               06BB  1304 MODIFIERS:
                   69    B5    06BB  1305          TSTW    TTY$W_IL_LEN(R9)          ; LENGTH MUST BE ZERO
                   F9    12    06BD  1306          BNEQ    GOBAD
                08 A9    D5    06BF  1307          TSTL    TTY$L_IL_RETADR(R9)       ; ALSO THE SECOND ADDRESS
                   F4    12    06C2  1308          BNEQ    GOBAD
          50    04 A9    D0    06C4  1309          MOVL    TTY$L_IL_ADR(R9),R0       ; GET THE MODIFIERS
          24 AB    50    C8    06C8  1310          BISL    R0,TTY$L_IS_MODIFY(R11)   ; SAVE THEM BUT DON'T DESTROY
                               06CC  1311                                            ; WHAT IS ALREADY THERE
                         CA    06CC  1312          BICL    #TRM$M_TM_CVTLOW!-        ; CLEAR ALL VALID BITS
                               06CD  1313                  TRM$M_TM_DSABLMBX!-
                               06CD  1314                  TRM$M_TM_NOECHO!-
                               06CD  1315                  TRM$M_TM_NOFILTR!-
                               06CD  1316                  TRM$M_TM_PURGE!-
                               06CD  1317                  TRM$M_TM_TIMED!-
                               06CD  1318                  TRM$M_TM_TRMNOECHO!-
                               06CD  1319                  TRM$M_TM_ESCAPE!-
                               06CD  1320                  TRM$M_TM_REFRESH!-
                               06CD  1321                  TRM$M_TM_NOEDIT!-
                               06CD  1322                  TRM$M_TM_R_JUST!-
                               06CD  1323                  TRM$M_TM_AUTO_TAB!-
                               06CD  1324                  TRM$M_TM_NORECALL,-
      50    0007FFC0 8F        06CD  1325                  R0
                   E3    12    06D3  1326          BNEQ    GOBAD                     ; MAKE SURE THAT THE MBZ BITS ARE Z
                         78    06D5  1327          ASHL    #TTY$V_ST_NOECHO-         ; Move function code and its
                               06D6  1328                  -IO$V_NOECHO,-            ; modifiers into bits 9-25 of
     50   24 AB    FD 8F       06D6  1329                  TTY$L_IS_MODIFY(R11),R0   ; a register.
      50   FFFFF3B7 8F   CA    06DB  1330          BICL    #^C<TTY$M_ST_NOECHO!-     ; Clear all bits except NOECHO
                               06E2  1331                  TTY$M_ST_NOFLTR!-         ; NOFLTR, and
                               06E2  1332                  TTY$M_ST_ESCAPE!-         ; ESCAPE
                               06E2  1333                  TTY$M_ST_REFRSH>,R0       ; REFRESH if specified.
                58    50    C8    06E2  1334          BISL    R0,R8                     ; SET THE NECESSARY BITS
          07 24 AB    0F    E1    06E5  1335          BBC     #TRM$V_TM_NOEDIT,TTY$L_IS_MODIFY(R11),10$; IF NOEDITING THEN MAKE IT
      58    00100000 8F   CA    06EA  1336          BICL    #TTY$M_ST_EDITING,R8      ; ...
                 FC4F    31    06F1  1337 10$:     BRW     ITEMLOOP                  ; AND GO BACK TO THE ITEM LIST LOOP
```

```
                          06F4   1339 ;++
                          06F4   1340 ;PICSTRNG
                          06F4   1341 ;
                          06F4   1342 ; DESCRIPTION:
                          06F4   1343 ;          VALIDATE THE LENGTH AND ADDRESS OF THE PICTURE STRING AND INCREASE
                          06F4   1344 ; THE BUFFER SIZE TO ACCOMIDATE IT.
                          06F4   1345 ;--
                          06F4   1346 PICSTRNG:
 01     04 AB    D1       06F4   1347          CMPL    TTY$L_IS_EDITMODE(R11),#TRM$K_EM_RDVERIFY; ARE WE A READ VERIFY READ
           BE    12       06F8   1348          BNEQ    GOBAD                         ; NO THEN ERROR OUT
 50     04 A9    D0       06FA   1349          MOVL    TTY$L_IL_ADR(R9),R0           ; GET THE ADDRESS OF THE PROMPT
    51     69    3C       06FE   1350          MOVZWL  TTY$W_IL_LEN(R9),R1           ; AND THE LENGTH
           17    13       0701   1351          BEQL    20$                           ; NO PROMPT THEN DO NOTHING
 28 AB     50    7D       0703   1352          MOVQ    R0,TTY$L_IS_PIC(R11)          ; SAVE THE PROMPT LENGTH AND ADDRESS
                          0707   1353 ASSUME   TTY$L_IS_PIC+4 EQ TTY$L_IS_PICLEN
    53     6B    D0       0707   1354          MOVL    TTY$L_IS_ACMODE(R11),R3       ; SETUP OUR ACCESS MODE
00000000'GF      16       070A   1355          JSB     G^EXE$PROBER                  ; CHECK THE ACCESS ON THE BUFFER
        03 50    E8       0710   1356          BLBS    R0,10$                        ; NO ERROR THEN CONTINUE
         FED0    31       0713   1357          BRW     ITMREADERR                    ; ELSE ERROR OUT
 57     2C AB    C0       0716   1358 10$:     ADDL    TTY$L_IS_PICLEN(R11),R7       ; ADD IN THE LENGTH
         FC26    31       071A   1359 20$:     BRW     ITEMLOOP                      ; CONTINUE ON NORMALY
```

```
                          071D  ;++
                          071D  ;PROMPT
                          071D  ;
                          071D  ; DESCRIPTION:
                          071D  ;       VALIDATE THE PROMPT ADDRESS AND LENGTH THEN SETUP THE NECESSARY STATE
                          071D  ; TO OUTPUT THE PROMPT
                          071D  ;--
                          071D  PROMPT:
       50    04 A9    D0  071D          MOVL    TTY$L_IL_ADR(R9),R0       ; GET THE ADDRESS OF THE PROMPT
          51    69    3C  0721          MOVZWL  TTY$W_IL_LEN(R9),R1       ; AND THE LENGTH
                30    13  0724          BEQL    20$                       ; NO PROMPT THEN DO NOTHING
       30 AB    50    7D  0726          MOVQ    R0,TTY$L_IS_PRM(R11)      ; SAVE THE PROMPT LENGTH AND ADDRESS
                          072A  ASSUME  TTY$L_IS_PRM+4 EQ TTY$L_IS_PRMLEN
       38 AB    51    D0  072A          MOVL    R1,TTY$L_IS_PRMBUF(R11)   ; THE BUFFER'S LENGTH
          53    6B    D0  072E          MOVL    TTY$L_IS_ACMODE(R11),R3 ; SETUP OUR ACCESS MODE
       00000000'GF    16  0731          JSB     G^EXE$PROBER              ; CHECK THE ACCESS ON THE BUFFER
             03 50    E8  0737          BLBS    R0,10$                    ; NO ERROR THEN CONTINUE
              FEA9    31  073A          BRW     ITMREADERR                ; ELSE ERROR OUT
          58    20    C8  073D  10$:    BISL    #TTY$M_ST_PROMPT,R8       ; SET THE PROMPTED READ BIT
40 AA    00000200 8F  C8  0740          BISL    #TTY$M_ST_EDITREAD,IRP$Q_TT_STATE(R10); MAKE SURE THE PROMPT IS ECHO
     0C 48 A5    0E   E0  0748          BBS     #TT2$V_FALLBACK,UCB$L_DEVDEPND2(R5),30$; HANDLE FALLBACK
          57    34 AB C0  074D  15$:    ADDL    TTY$L_IS_PRMLEN(R11),R7 ; ADD IN THE LENGTH
     00 2A AA    01   E2  0751          BBSS    #IRP$V_FUNC,IRP$W_STS(R10),20$; RESET TRANSFER DIRECTION
              FBEA    31  0756  20$:    BRW     ITEMLOOP                  ; CONTINUE ON NORMALY
                          0759
                59    DD  0759  30$:    PUSHL   R9                        ; SAVE R9 OVER THE CALL
       51    30 AB    D0  075B          MOVL    TTY$L_IS_PRM(R11),R1      ; RESTORE ADDRESS AND LENGTH
       50    34 AB    D0  075F          MOVL    TTY$L_IS_PRMLEN(R11),R0 ; AND THE LENGTH
              0600    30  0763          BSBW    ADDFALL                   ; CALCULATE THE ADDITIONAL
                          0766                                            ; CHARACER COUNT OF FALLBACK
       34 AB    59    C0  0766          ADDL    R9,TTY$L_IS_PRMLEN(R11)  ;ADD IN THE COUNT
             59 8ED0      076A          POPL    R9                        ; RESTORE R9
             DE   11      076D          BRB     15$                       ;
```

```
                              076F  1395 ;++
                              076F  1396 ; TERM
                              076F  1397 ;
                              076F  1398 ; DESCRIPTION:
                              076F  1399 ;        SETUP TERMINATOR MASK. AND VALIDATE IT.
                              076F  1400 ; THE TERMINATOR MASK SHORT FORM IS SPECIFYED BY SETTING THE LENGTH
                              076F  1401 ; TO ZERO AND THE BIT MASK GOES IN ADR.  THE LONG FORM LENGTH GOES
                              076F  1402 ; IN LENGTH AND THE ADDRESS IS PLACED IN ADR.
                              076F  1403 ;--
                              076F  1404 TERM:
          50    04 A9     D0  076F  1405         MOVL    TTY$L_IL_ADR(R9),R0       ; GET THE TERMINATOR ADDRESS
             51    69     3C  0773  1406         MOVZWL  TTY$W_IL_LEN(R9),R1       ; AND THE LENGTH
                   0B     12  0776  1407         BNEQ    10$                       ; SHORT FORM NO THEN HANDLE LONG FORM
          44 AB    04     9A  0778  1408         MOVZBL  #4,TTY$L_IS_TERMLEN(R11); SHORT FORM LENGTH
       40 AB    04 A9     DE  077C  1409         MOVAL   TTY$L_IL_ADR(R9),TTY$L_IS_TERM(R11); AND THE ADDRESS OF THE MASK
                   13     11  0781  1410         BRB     20$
                              0783  1411
          40 AB    50     7D  0783  1412 10$:    MOVQ    R0,TTY$L_IS_TERM(R11)     ; SAVE THE TERMINATOR MASK AND LENGTH
                              0787  1413 ASSUME  TTY$L_IS_TERM+4 EQ TTY$L_IS_TERMLEN
             53    6B     D0  0787  1414         MOVL    TTY$L_IS_ACMODE(R11),R3 ; GET THE ACCESS MODE
           00000000'GF    16  078A  1415         JSB     G^EXE$PROBER              ; AND PROBE THE TERMINATOR MASK
                03 50     E8  0790  1416         BLBS    R0,20$
                 FE50     31  0793  1417         BRW     ITMREADERR                ; NO GOOD THENB TELL HIM SO
                              0796  1418
             57    20     C0  0796  1419 20$:    ADDL    #32,R7                    ; 16 BYTES FOR TERMINATOR MASK
   58    01000000 8F      C8  0799  1420         BISL    #TTY$M_ST_TERMNORM,R8     ; NON-STANDARD TERMINATOR
                              07A0  1421                                           ; MASK THEN LET THE TERMINATORS THRU
                 FBA0     31  07A0  1422 30$:    BRW     ITEMLOOP                  ; IF OK THEN GO ON TO THE NEXT ONE
```

```
                         07A3  1424 ;++
                         07A3  1425 ; TIMEOUT
                         07A3  1426 ;
                         07A3  1427 ; DESCRIPTION
                         07A3  1428 ;           VALIDATES ARGUMENTS, SAVES TIMEOUT VALUE AND SETS TIMED FUNCTION
                         07A3  1429 ; BITS.
                         07A3  1430 ;--
                         07A3  1431 TIMEOUT:
             69    B5    07A3  1432             TSTW    TTY$W_IL_LEN(R9)          ; LENGTH MUST BE ZERO
             12    12    07A5  1433             BNEQ    20$
          08 A9    D5    07A7  1434             TSTL    TTY$L_IL_RETADR(R9)       ; ALSO THE SECOND ADDRESS
             0D    12    07AA  1435             BNEQ    20$
  50 AB   04 A9    D0    07AC  1436             MOVL    TTY$L_IL_ADR(R9),TTY$L_IS_TIMEOUT(R11); GET THE TIMEOUT VALUE
  00 24 AB   07    E2    07B1  1437             BBSS    #TRM$V_TM_TIMED,TTY$L_IS_MODIFY(R11),10$; SET THE TIMEOUT MODIFIER
           FB8A    31    07B6  1438 10$:        BRW     ITEMLOOP
           FE23    31    07B9  1439 20$:        BRW     BDPRMERR
```

```
                    07BC        .SBTTL  TTY$FDTWRITE - Function decision routine for terminal writes
                    07BC
                    07BC  ;++
                    07BC  ; TTY$FDTWRITE - FUNCTION DECISION ROUTINE FOR TERMINAL WRITE FUNCTIONS
                    07BC  ;
                    07BC  ; FUNCTIONAL DESCRIPTION:
                    07BC  ;
                    07BC  ; THIS ROUTINE IS THE FUNCTION DECISION ROUTINE FOR TERMINAL WRITE FUNCTIONS.
                    07BC  ;
                    07BC  ; THE QIO PARAMETERS FOR TERMINAL WRITES ARE:
                    07BC  ;
                    07BC  ;         P1 = ADDRESS OF THE BUFFER
                    07BC  ;         P2 = SIZE OF THE BUFFER
                    07BC  ;         P3 = UNUSED
                    07BC  ;         P4 = CARRIAGE CONTROL SPECIFIER (SEE EXE$CARRIAGE)
                    07BC  ;
                    07BC  ; THE FUNCTION PARAMETERS ARE VALIDATED AND IF CORRECT, a write packet
                    07BC  ; that points to the IRP is sent to the terminal driver's write start
                    07BC  ; I/O routine (ALTSTART in the DDT).
                    07BC  ;
                    07BC  ;
                    07BC  ;         IRP$W_BOFF CONTAINS THE QUOTA FOR THIS I/O
                    07BC  ;         IRP$W_BCNT CONTAINS THE TRANSFER COUNT
                    07BC  ;         IRP$W_FUNC IS SET FOR A FAST CASE ON FUNCTION TYPE
                    07BC  ;
                    07BC  ; FOR IO$_WRITEPBLK OR IO$M_NOFORMAT, TTY$V_ST_WRTALL IS SET TO PREVENT
                    07BC  ; FORMATTING OF THE DATA.
                    07BC  ;
                    07BC  ;         If IO$_REFRESH is specified, this routine sets TTY$V_ST_REFRSH
                    07BC  ;         to refresh a delayed read when the write completes.
                    07BC  ;
                    07BC  ; INPUTS:
                    07BC  ;
                    07BC  ;         R3 = I/O PACKET
                    07BC  ;         R4 = PCB OF PROCESS
                    07BC  ;         R5 = UCB
                    07BC  ;         R6 = ASSIGNED CCB
                    07BC  ;         R7 = FUNCTION CODE
                    07BC  ;         AP = ADDRESS OF FIRST USER QIO PARAMETER
                    07BC  ;
                    07BC  ; OUTPUTS:
                    07BC  ;
                    07BC  ;         IF THE I/O IS IN ERROR THEN IT IS COMPLETED BY "EXE$ABORTIO".
                    07BC  ;         If the I/O is valid, then the address of the buffered block is
                    07BC  ;         loaded into R3, and the block queued to EXE$ALTQUEPKT.
                    07BC  ;
                    07BC  ; COMPLETION CODES:
                    07BC  ;
                    07BC  ;         SS$_ACCVIO - ACCESS VIOLATION ON BUFFER ( FROM "EXE$WRTCHK" )
                    07BC  ;         SS$_INSFMEM - INSUFFICIENT MEMORY FOR REQUEST
                    07BC  ;         SS$_EXQUOTA - BUFFERED I/O QUOTA EXCEEDED
                    07BC  ;--
                    07BC
                    07BC  TTY$FDTWRITE::                                      ;
        56   6C  D0 07BC          MOVL    P1(AP),R6                           ; GET USER BUFFER VIRTUAL ADDRESS
          3C A3  D4 07BF          CLRL    IRP$B_CARCON(R3)                    ; ASSUME NO CARRIAGE CONTROL SPECIFIER
0B  57  06   00  ED 07C2          CMPZV   #IRP$V_FCODE,#IRP$S_FCODE,R7,#IO$_WRITEPBLK; WRITE PHYSICAL BLOCK?
```

```
                  0B  13 07C7  1498         BEQL    10$                      ; IF EQL THEN YES
      3C A3    0C AC  D0 07C9  1499         MOVL    P4(AP),IRP$B_CARCON(R3)  ; GET CARRIAGE CONTROL SPECIFIER
         00000000'GF  16 07CE  1500         JSB     G^EXE$CARRIAGE           ; CONVERT THE CARRIAGE CONTROL
      58    3C A3     9A 07D4  1501 10$:    MOVZBL  IRP$B_CARCON(R3),R8      ; CALC NUMBER OF EXTRA PLACES
      52    3E A3     9A 07D8  1502         MOVZBL  IRP$B_CARCON+2(R3),R2    ; NEEDED FOR CONTROL
            58    52  C0 07DC  1503         ADDL    R2,R8                    ;
            50    56  D0 07DF  1504         MOVL    R6,R0                    ; SET UP FOR WRITE CHECK CALL
      57    04 AC     3C 07E2  1505         MOVZWL  P2(AP),R7                ; GET TRANSFER SIZE
            51    57  D0 07E6  1506         MOVL    R7,R1                    ; COPY TRANSFER SIZE
                  06  13 07E9  1507         BEQL    12$                      ; SKIP CHECK IF ZERO
         00000000'GF  16 07EB  1508         JSB     G^EXE$WRITECHK           ; CHECK BUFFER ACCESS
                         07F1  1509 ;
                         07F1  1510 ; INIT STATE FIELD
                         07F1  1511 ;                                        ; NO RETURN MEANS NO ACCESS
                         07F1  1512 12$:
               40 A3  7C 07F1  1513         CLRQ    IRP$Q_TT_STATE(R3)       ; INIT STATE REGION
             0080 8F  3C 07F4  1514         MOVZWL  #<TTY$M_ST_WRITE>,-
               40 A3     07F8  1515                 IRP$Q_TT_STATE(R3)       ; INIT WRITE FUNCTION
                         07FA  1516
                         07FA  1517 ;
                         07FA  1518 ; SET WRITE PASSALL STATE FOR WRTPASSALL
                         07FA  1519 ;
    08 20 A3    08    E0 07FA  1520         BBS     #IO$V_NOFORMAT,IRP$W_FUNC(R3),15$; BR IF NO FORMAT SPECIFIED
0B    20 A3    06    00 ED 07FF  1521         CMPZV   #IRP$V_FCODE,#IRP$S_FCODE,IRP$W_FUNC(R3),#IO$_WRITEPBLK; PASSALL WRI
                  1D  12 0805  1522         BNEQ    25$                      ; No passall, branch forward.
                         0807  1523
                         0807  1524 15$:
                  04  E2 0807  1525         BBSS    #TTY$V_ST_WRTALL,-       ; Set no format mode for
            00 44 A3     0809  1526                 IRP$Q_TT_STATE+4(R3),17$        ; write.
    13 48 A5    0E    E0 080C  1527 17$:    BBS     #TT2$V_FALLBACK,UCB$L_DEVDEPND2(R5),25$; NOT DOING FALLBACK THEN DON
                         0811  1528
                  38  BB 0811  1529 20$:    PUSHR   #^M<R3,R4,R5>            ; SAVE SOME REGISTERS
      51    34 A841   9E 0813  1530         MOVAB   TTY$L_WB_DATA+4(R8)[R1],R1; ADD HEADER TO REQUEST AND CARRIAGE CONTR
                         0818  1531 ;
                         0818  1532 ; CHECK BUFFERED I/O QUOTA
                         0818  1533 ;
         00000000'GF  16 0818  1534         JSB     G^EXE$BUFFRQUOTA         ; CHECK QUOTA
               06 50  E8 081E  1535         BLBS    R0,30$                   ; Branch forward on success.
                00B3  31 0821  1536         BRW     105$                     ; Otherwise, branch to error.
                00E5  31 0824  1537 25$:    BRW     200$                     ; jump to the fallback logic
                         0827  1538 ;
                         0827  1539 ; Allocate the buffer.
                         0827  1540 ;
                         0827  1541
                         0827  1542 30$:
         00000000'GF  16 0827  1543         JSB     G^EXE$ALLOCBUF           ; Allocate buffered I/O block.
               03 50  E8 082D  1544         BLBS    R0,40$                   ; Branch forward on success.
                00A4  31 0830  1545         BRW     105$                     ; Otherwise, branch to error
                         0833  1546
                         0833  1547 40$:
                         0833  1548                                          ; exit.
            53    6E  D0 0833  1549         MOVL    (SP),R3                  ; RESTORE PACKET ADDRESS
      2C A3    52     D0 0836  1550         MOVL    R2,IRP$L_SVAPTE(R3)      ; SAVE BLOCK ADDRESS IN PACKET
      32 A3    58     A0 083A  1551         ADDW    R8,IRP$W_BCNT(R3)        ; ADJUST TRANSFER SIZE FOR CARRIAGE CONTROL
                         083E  1552 ;
                         083E  1553 ; ADJUST QUOTA
                         083E  1554 ;
      58   0080 C4   D0  083E  1555          MOVL    PCB$L_JIB(R4),R8           ; GET JIB ADDRESS
             51 51   3C  0843  1556          MOVZWL  R1,R1                      ; CONVERT COUNT TO LONGWORD
       20 A8    51   C2  0846  1557          SUBL    R1,JIB$L_BYTCNT(R8)        ; ADJUST BUFFERED I/O QUOTA
       30 A3    51   B0  084A  1558          MOVW    R1,IRP$W_BOFF(R3)          ; SAVE BLOCK SIZE AS QUOTA
                         084E  1559
                         084E  1560 ;
                         084E  1561 ;        MARK PACKET AS TERMINAL I/O
                         084E  1562 ;
         57   04     D1  084E  1563          CMPL    #4,R7                      ; No priority boost for programs
              06     18  0851  1564          BGEQ    45$                        ; that do small io
 2A A3   0200 8F     A8  0853  1565          BISW    #IRP$M_TERMIO,IRP$W_STS(R3)     ; SET FLAG FOR GREATER PRIORITY
                         0859  1566 45$:                                        ; INCREMENTS UPON COMPLETION
                         0859  1567
                         0859  1568 ;
                         0859  1569 ; SET UP THE BLOCK
                         0859  1570 ;
              30     9A  0859  1571          MOVZBL  #DYN$C_TWP,-               ; Insert block type.
            0A A2        085B  1572                  TTY$B_WB_TYPE(R2)
       24 A2  53     D0  085D  1573          MOVL    R3,TTY$L_WB_IRP(R2)        ; Insert IRP address.
            30 A2    9E  0861  1574          MOVAB   TTY$L_WB_DATA(R2),-        ; Insert start of data address
            1C A2        0864  1575                  TTY$L_WB_NEXT(R2)          ; in "next character" field.
         52   30     C0  0866  1576          ADDL    #TTY$C_WB_DATA,R2          ; POINT TO DATA
                         0869  1577 ;
                         0869  1578 ; INSERT INITIAL CARRIAGE CONTROL
                         0869  1579 ;
      58    3C A3    9E  0869  1580          MOVAB   IRP$B_CARCON(R3),R8        ; INSERT THE CHARACTERS
              70     10  086D  1581          BSBB    110$                       ;
                         086F  1582 ;
                         086F  1583 ; CHECK FOR UPPERCASE  AND FALLBACK
                         086F  1584 ; DON'T WORRIE ABOUT THEM IF WE ARE IN WRITEALL MODE
                         086F  1585 ;
              0E     E0  086F  1586          BBS     #TT2$V_FALLBACK,-
         14 48 A5        0871  1587                  UCB$L_DEVDEPND2(R5),47$         ; NOT DOING FALLBACK THEN DON'T WORR
              04     E0  0874  1588          BBS     #TTY$V_ST_WRTALL,-              ; If no format mode
         12 44 A3        0876  1589                  IRP$Q_TT_STATE+4(R3),50$        ; no translate.
              00     E0  0879  1590          BBS     #TT$V_PASSALL,-
         0D 44 A5        087B  1591                  UCB$L_DEVDEPEND(R5),50$ ; CHECK FOR PASSALL OR PASTHRU
              12     E0  087E  1592          BBS     #TT2$V_PASTHRU,-
         08 48 A5        0880  1593                  UCB$L_DEVDEPND2(R5),50$ ; ...
              07     E0  0883  1594          BBS     #TT$V_LOWER,-              ; CHECK IF LOWER CASE ALLOWED
         03 44 A5        0885  1595                  UCB$L_DEVDEPEND(R5),50$
            006D     31  0888  1596 47$:     BRW     150$
                         088B  1597 ;
                         088B  1598 ; COPY USER DATA TO BUFFER
                         088B  1599 ;
      62    66  57   28  088B  1600 50$:     MOVC3   R7,(R6),(R2)               ; MOVE THE DATA
            52  53   D0  088F  1601          MOVL    R3,R2                      ; COPY CURRENT END OF DATA
                         0892  1602 ;
                         0892  1603 ; THE USER DATA IS COPIED -- ADD TRAILING CARRIAGE RETURN IF NECESS.
                         0892  1604 ;
              38     BA  0892  1605 60$:     POPR    #^M<R3,R4,R5>              ; RESTORE REGISTERS
      58    3E A3    9E  0894  1606          MOVAB   IRP$B_CARCON+2(R3),R8      ;
              45     10  0898  1607          BSBB    110$                       ; INSERT CHARACTERS
                         089A  1608
                         089A  1609 ;
                         089A  1610 ; If the write function specified IO$_REFRESH, set the appropriate bit
                         089A  1611 ; position for the UCB state bits.
```

```
                       089A  ;
                       089A
                       089A  95$:
             0D    E1  089A          BBC     #IO$V_REFRESH,-                 ; Check for REFRESH bit.
       05 20 A3        089C                  IRP$W_FUNC(R3),100$             ; If REFRESH is not specified,
                                                                             ; just branch forward.
             0A    E2  089F          BBSS    #TTY$V_ST_REFRSH,-              ; Otherwise, set the refresh
       00 44 A3        08A1                  IRP$Q_TT_STATE+4(R3),100$              ; bit for the state longword.
                       08A4
                       08A4  ;
                       08A4  ; COMPLETE THE WRITE OPERATION
                       08A4  ;
    51    2C A3    D0  08A4  100$:   MOVL    IRP$L_SVAPTE(R3),R1             ; GET BLOCK ADDRESS
       20 A1  52   D0  08A8          MOVL    R2,TTY$L_WB_END(R1)             ; INSERT ADDRESS OF DATA END
20 A3    06   00   01  F0  08AC      INSV    #TTY$C_FC_WRITE,#IRP$V_FCODE,#IRP$S_FCODE,IRP$W_FUNC(R3);
          32 A3    B0  08B2          MOVW    IRP$W_BCNT(R3),-                ; Move the character count
          2A A1        08B5                  TTY$W_WB_BCNT(R1)               ; into the write packet.
       00A0 C5     D5  08B7          TSTL    UCB$L_TL_PHYUCB(R5)             ; Test for disconnected LUCB
             05    13  08BB          BEQL    101$                            ; Always que 1/2 duplex if so
    06 44 A5  14   E1  08BD          BBC     #TT$V_HALFDUP, -                ; branch if full duplex
                       08C2                  UCB$L_DEVDEPEND(R5), -          ; if half duplex, call normal
                       08C2                  102$                            ; tty$startio entry point
       00000000'GF 17  08C2  101$:   JMP     G^EXE$QIODRVPKT                 ; R3/addr of write IRP
                       08C8  102$:
          53  51   D0  08C8          MOVL    R1,R3                           ; Set up write block address.
       00000000'GF 16  08CB          JSB     G^EXE$ALTQUEPKT                 ; Queue packet to driver's write
                       08D1                                                  ; STARTIO routine.
       00000000'GF 17  08D1          JMP     G^EXE$QIORETURN                 ; Return to requesting process.
                       08D7  ;
                       08D7  ; ERROR IN PROCESSING
                       08D7  ;
             38    BA  08D7  105$:   POPR    #^M<R3,R4,R5>                   ; RESTORE REGISTERS
       00000000'GF 17  08D9          JMP     G^EXE$ABORTIO                   ; ABORT THE I/O
                       08DF  ;
                       08DF  ; SUBROUTINE TO INSERT PRE/SUF CARRIAGE CONTROL
                       08DF  ;
       50    68    9A  08DF  110$:   MOVZBL  (R8),R0                         ; GET NUMBER OF CHARACTERS
             13    13  08E2          BEQL    130$                            ; IF EQL THEN NONE
    82    01 A8    90  08E4          MOVB    1(R8),(R2)+                     ; INSERT CHARACTER
             0D    12  08E8          BNEQ    130$                            ; IF NEQ THEN DONE
    FF A2    0D    90  08EA          MOVB    #TTY$C_CR,-1(R2)                ; INSERT CARRIAGE RETURN TO START
          32 A3    B6  08EE          INCW    IRP$W_BCNT(R3)                  ; INCREASE BYTE COUNT FOR CR
       82    0A    90  08F1  120$:   MOVB    #TTY$C_LF,(R2)+                 ; INSERT LINE FEEDS
          FA 50    F5  08F4          SOBGTR  R0,120$                         ; UNTIL DONE
             05        08F7  130$:   RSB                                     ;
                       08F8
                       08F8  ;
                       08F8  ; TRANSLATE TO UPPERCASE
                       08F8  ;
             58    D4  08F8  150$:   CLRL    R8
             07    E0  08FA          BBS     #TT$V_LOWER,-                   ; CHECK IF LOWER CASE ALLOWED
       03 44 A5        08FC                  UCB$L_DEVDEPEND(R5),155$
          58  01   D0  08FF          MOVL    #1,R8
    53    57  59   C1  0902  155$:   ADDL3   R9,R7,R3                        ; CALCULATE THE NEW STRING LENGTH
           039A    30  0906          BSBW    MOVE_TRANSLATE
           FF86    31  0909          BRW     60$
                       090C  ;
                       090C  ; Figure out how many characters will be added to this write
```

```
                         090C  1669 ; for fallback presentation
                         090C  1670 ;
                         090C  1671
      10 48 A5   0E  E1  090C  1672 200$:     BBC     #TT2$V_FALLBACK,UCB$L_DEVDEPND2(R5),215$; NOT DOING FALLBACK THEN DO
                 0F  BB  0911  1673           PUSHR   #^M<R0,R1,R2,R3>              ; save the registers destroyed by the
                         0913  1674                                                 ; scan
            50   57  D0  0913  1675           MOVL    R7,R0                         ; SETUP THE LENGTH
            51   56  D0  0916  1676           MOVL    R6,R1                         ; AND THE ADDRESS
               044A  30  0919  1677           BSBW    ADDFALL                       ; ADD IN THE FALLBACK COUNT
                 0F  BA  091C  1678           POPR    #^M<R0,R1,R2,R3>              ; RESTORE THE REGISTERS
            51   59  C0  091E  1679           ADDL    R9,R1                         ; AND GET THE EXTRA SPACE
               FEED  31  0921  1680 215$:     BRW     20$
                         0924  1681
                         0924  1682
```

```
                                   0924          .SBTTL  TTY$FDTSETM -- FUNCTION DECISION ROUTINE FOR TERMINAL SET MODE
                                   0924  ;++
                                   0924  ; TTY$FDTSETM - FUNCTION DECISION ROUTINE FOR TERMINAL SET MODE FUNCTIONS
                                   0924  ;
                                   0924  ; FUNCTIONAL DESCRIPTION:
                                   0924  ;
                                   0924  ; THIS ROUTINE IS THE FUNCTION DECISION ROUTINE FOR TERMINAL SET MODE FUNCTIONS.
                                   0924  ; THERE ARE TWO BASIC FUNCTIONS -- SET UP FOR CONTROL C AND SET MODE.
                                   0924  ;
                                   0924  ; THE FUNCTION CODE IS SET FOR A FAST CASE ON TYPE
                                   0924  ;
                                   0924  ; INPUTS:
                                   0924  ;
                                   0924  ;        R3 = I/O PACKET ADDRESS
                                   0924  ;        R4 = PCB ADDRESS OF CURRENT PROCESS
                                   0924  ;        R5 = UCB ADDRESS
                                   0924  ;        R6 = CCB ADDRESS FOR ASSIGNED UNIT
                                   0924  ;        AP = ADDRESS OF ARGUMENT LIST AT USER PARAMETERS
                                   0924  ;
                                   0924  ; OUTPUTS:
                                   0924  ;
                                   0924  ;        THE FUNCTION IS COMPLETED HERE BY "EXE$FINISHIO".
                                   0924  ;
                                   0924  ; IMPLICIT OUTPUTS:
                                   0924  ;
                                   0924  ;        R3,R5 ARE PRESERVED.
                                   0924  ;--
                                   0924  TTY$FDTSETM::
  20 A3    06    00    02    F0    0924          INSV    #TTY$C_FC_SETM,#IRP$V_FCODE,#IRP$S_FCODE,IRP$W_FUNC(R3);
        44 A3    48 A5    D0    092A          MOVL    UCB$L_DEVDEPND2(R5),IRP$Q_TT_STATE+4(R3)         ;INIT DEFAULT
                       06    11    092F          BRB     SET_COMMON
```

```
                          0931  1716          .SBTTL  TTY$FDTSETC - FUNCTION DECISION ROUTINE FOR TERMINAL SET CHARS
                          0931  1717  ;++
                          0931  1718  ; TTY$FDTSETC - FUNCTION DECISION ROUTINE FOR TERMINAL SET CHARACTERISTICS FUNCTIONS
                          0931  1719  ;
                          0931  1720  ; FUNCTIONAL DESCRIPTION:
                          0931  1721  ;
                          0931  1722  ; THIS ROUTINE IS THE FUNCTION DECISION ROUTINE FOR TERMINAL SET MODE FUNCTIONS.
                          0931  1723  ; THERE ARE TWO BASIC FUNCTIONS -- SET UP FOR CONTROL Y AND SET MODE.
                          0931  1724  ;
                          0931  1725  ; INPUTS:
                          0931  1726  ;
                          0931  1727  ;       R3 = I/O PACKET ADDRESS
                          0931  1728  ;       R4 = PCB ADDRESS OF CURRENT PROCESS
                          0931  1729  ;       R5 = UCB ADDRESS
                          0931  1730  ;       R6 = CCB ADDRESS FOR ASSIGNED UNIT
                          0931  1731  ;       AP = ADDRESS OF ARGUMENT LIST AT USER PARAMETERS
                          0931  1732  ;
                          0931  1733  ; OUTPUTS:
                          0931  1734  ;
                          0931  1735  ;       THE FUNCTION IS COMPLETED HERE BY "EXE$FINISHIO".
                          0931  1736  ;       OR BY QUEUING IT TO FOR FOLLOW ON PROCESSING
                          0931  1737  ;       BY TTY$TRSTP.
                          0931  1738  ;
                          0931  1739  ; IMPLICIT OUTPUTS:
                          0931  1740  ;
                          0931  1741  ;       R3,R5 ARE PRESERVED.
                          0931  1742  ;--
                          0931  1743  TTY$FDTSETC::
 20 A3   06   00   03   F0 0931  1744          INSV    #TTY$C_FC_SETC,#IRP$V_FCODE,#IRP$S_FCODE,IRP$W_FUNC(R3);
                          0937  1745
                          0937  1746  SET_COMMON:
           09   06   EA    0937  1747          FFS     #IO$V_MAINT,#9,-
         51   20 A3        093A  1748                  IRP$W_FUNC(R3),R1       ; GET PRIMARY MODIFIER BIT
                          093D  1749          CASE    R1,TYPE=B,LIMIT=#6,<-   ; AND VECTOR TO SERVICE ROUTINE
                          093D  1750          SET_MAINT,-
                          093D  1751          SET_CTRLY,-
                          093D  1752          SET_CTRLC,-
                          093D  1753          SET_HANGUP,-
                          093D  1754          SET_OUTBAND,-
                          093D  1755          SET_CONNECT,-
                          093D  1756          SET_DISCONNECT,-
                          093D  1757          SET_PID,-
                          093D  1758          SET_BRDCST>
                          0953  1759
```

```
                               0953  1761 ;
                               0953  1762 ; PROCESS SET MODE OR CHARACTERISTICS
                               0953  1763 ;
                               0953  1764 SET:
                   018F   30   0953  1765          BSBW     GET_PARAMS                   ; GET USER PARAMETERS
           51    38 A3    DE   0956  1766          MOVAL    IRP$L_MEDIA(R3),R1           ; GET ADDRESS OF STORED USER DATA
                 02 A1    B5   095A  1767          TSTW     2(R1)                        ; PAGE WIDTH 0?
                    08    15   095D  1768          BLEQ     BAD_SET                      ; 0 IS BAD WIDTH
      02 A1    01FF 8F    B1   095F  1769          CMPW     #TTY$C_MAXPAGWID,2(R1)       ; RESTRICT WIDTH
                    09    18   0965  1770          BGEQ     GOOD_SET                     ; FALL THROUGH IF > MAX
                               0967  1771
              50    14    3C   0967  1772 BAD_SET:MOVZWL   #SS$_BADPARAM,R0             ; SET ERROR
          00000000'GF     17   096A  1773          JMP      G^EXE$ABORTIO                ; RETURN ON ERROR
                               0970  1774
                               0970  1775 GOOD_SET:
        4C A3    08 AC    B0   0970  1776          MOVW     P3(AP),IRP$W_TT_PRMPT(R3)  ; GET THE SPEED PARAMETER
        4E A3    0C AC    B0   0975  1777          MOVW     P4(AP),IRP$W_TT_PRMPT+2(R3); SAVE FILLS
      009C C3    10 AC    D0   097A  1778          MOVL     P5(AP),IRP$L_VAL5(R3)      ; SAVE PARITY
 51    009C C3   FFFFFFF0 8F    CB   0980  1779          BICL3    #^C<^X0F>,IRP$L_VAL5(R3),R1; GET THE FRAME SIZE
                    14    13   098A  1780          BEQL     5$                           ; NOTHING THEN QUEUE THE PACKET
              51    05    C2   098C  1781          SUBL     #5,R1                        ; IS THIS IN RANGE
                    D6    19   098F  1782          BLSS     BAD_SET                      ; NO THEN BAD PARAMETER
              03    51    D1   0991  1783          CMPL     R1,#3                        ; MUST BE BETWEEN 8 AND 5
                    D1    14   0994  1784          BGTR     BAD_SET                      ; NOT THERE THEN EXIT
         009C C3    0F    CA   0996  1785          BICL     #^X0F,IRP$L_VAL5(R3)         ; CLEAN THE OLD BITS
         009C C3    51    C8   099B  1786          BISL     R1,IRP$L_VAL5(R3)            ; AND SET IN THE NEW ONES
                               09A0  1787 5$:
          00000000'GF     17   09A0  1788 QPKT:    JMP      G^EXE$QIODRVPKT              ; QUEUE PACKET TO DRIVER
                               09A6  1789
```

```
                         09A6  1791
                         09A6  1792 ; STORE BROADCAST MASK IN LOGICAL UCB
                         09A6  1793
                         09A6  1794 SET_BRDCST:
            013C    30   09A6  1795          BSBW    GET_PARAMS                        ; GET USER ARGUMENTS
    00A8 C5   61    7D   09A9  1796          MOVQ    (R1),UCB$Q_TL_BRKTHRU(R5)         ; SAVE SPECIFIED MASK
            00ED    31   09AE  1797          BRW     SET_DONE                          ; AND EXIT NORMALY
```

```
                              09B1  1799 ;
                              09B1  1800 ; Fill in the ucb fields and whatever else is needed to initiate
                              09B1  1801 ; a connect to a vertual terminal
                              09B1  1802 ;
                              09B1  1803
                              09B1  1804 SET_CONNECT:
                    0009  30  09B1  1805         BSBW    GET_LUCB
  20 A3    06    00   07  F0  09B4  1806         INSV    #TTY$C_FC_CONNECT,#IRP$V_FCODE,#IRP$S_FCODE,IRP$W_FUNC(R3);
                    FFE3  31  09BA  1807         BRW     QPKT                            ; QUEUE PACKET FOR FOLLOW ON PROCESSING
                              09BD  1808
                              09BD  1809
                              09BD  1810 ;       LOOK UP LUCB NAME IN IO DATA BASE
                              09BD  1811 ;       AND VALIDATE ACCESS TO IT FROM COMMAND CHANNEL
                              09BD  1812
                              09BD  1813 GET_LUCB:
                      53  DD  09BD  1814         PUSHL   R3                                      ; SAVE IRP ADDRESS
             00000000'GF  16  09BF  1815         JSB     G^SCH$IOLOCKW                           ; INTERLOCK IO DATA BASE
                    011D  30  09C5  1816         BSBW    GET_PARAMS                              ; PROBE BUFFER DESCRIPTOR
             00000000'GF  16  09C8  1817         JSB     G^IOC$SEARCHDEV                         ; GO FIND DEVICE UCB ADDRESS
                              09CE  1818                                                 ; IF SUCCESS R1= TARGET UCB
                              09CE  1819                                                 ; R2 = DDB OF TARGET UCB
                   14 50  E8  09CE  1820         BLBS    R0,10$                                  ; UCB FOUND?
                              09D1  1821
                              09D1  1822         ;ERROR EXIT
                              09D1  1823 5$:
                      50  DD  09D1  1824         PUSHL   R0                                      ; SAVE ERROR STATUS
             00000000'GF  16  09D3  1825         JSB     G^SCH$IOUNLOCK                          ; INTERLOCK IO DATA BASE
                    50 8ED0   09D9  1826         POPL    R0                                      ; RESTORE ERROR STATUS
                    53 8ED0   09DC  1827         POPL    R3                                      ; RESTORE IRP
             00000000'GF  17  09DF  1828         JMP     G^EXE$ABORTIO                           ; ERROR
                              09E5  1829 10$:
          50   0144 8F    3C  09E5  1830         MOVZWL  #SS$_IVDEVNAM,R0                        ; ASSUME INVALID DEVICE
       00000000'EF    52  D1  09EA  1831         CMPL    R2,VT$DDB                               ; VERIFY TARGET LUCB ON DETACHED  DD
                      DE  12  09F1  1832         BNEQ    5$                                      ; NO, SO ABORT
                              09F3  1833         DSBINT  #IPL$_SYNCH                             ; INTERLOCK WITH DRIVER FORK
             52    2C A1  3C  09F9  1834         MOVZWL  UCB$L_PID(R1),R2                        ; GET PID OF OWNER PROCESS OF LUCB
        50   00000000'GF  D0  09FD  1835         MOVL    G^SCH$GL_PCBVEC,R0                      ; GET ADDRESS OF PCB ARRAY
             52    6042   D0  0A04  1836         MOVL    (R0)[R2],R2                             ; GET PCB ADDRESS OF LUCB OWNER
    00BC C4    00BC C2    D1  0A08  1837         CMPL    PCB$L_UIC(R2),PCB$L_UIC(R4)             ; UIC MATCH (TARGET LUCB : COMMAND C
                      24  12  0A0F  1838         BNEQ    15$                                     ; NO , PRIV ERROR
        50     00A0 C5    D0  0A11  1839 12$:    MOVL    UCB$L_TL_PHYUCB(R5),R0                  ; GET ADDRESS OF PHYUCB OF COMMAND C
                      25  13  0A16  1840         BEQL    17$                                     ; ERROR, CURRENTLY DETACHED
               00A0 C1    D5  0A18  1841         TSTL    UCB$L_TL_PHYUCB(R1)                     ; CHECK ADDRESS OF PHYUCB OF TARGET
                      1F  12  0A1C  1842         BNEQ    17$                                     ; ERROR IF CURRENTLY CONNECTED
         0084 C0      51  D0  0A1E  1843         MOVL    R1,UCB$L_PDT(R0)                        ; SAVE ADDRESS TARGET LUCB IN COMMAN
         0084 C1      50  D0  0A23  1844         MOVL    R0,UCB$L_PDT(R1)                        ; SHOW CONNECT PENDING ON TARGET LUC
                              0A28  1845         ENBINT
             00000000'GF  16  0A2B  1846         JSB     G^SCH$IOUNLOCK                          ; INTERLOCK IO DATA BASE
                    53 8ED0   0A31  1847         POPL    R3
                          05  0A34  1848         RSB
                              0A35  1849 15$:
                              0A35  1850         ENBINT
             50       24  3C  0A38  1851         MOVZWL  #SS$_NOPRIV,R0
                      94  11  0A3B  1852         BRB     5$
                              0A3D  1853 17$:
                              0A3D  1854         ENBINT
        50     0840 8F    3C  0A40  1855         MOVZWL  #SS$_DEVALLOC,R0
```

```
                     8A   11  0A45  1856          BRB      5$
```

```
                         0A47  1858
                         0A47  1859 ;        ENABLE CONTROL C/Y
                         0A47  1860
                         0A47  1861 SET_CTRLC:
      57   0094 C5   9E  0A47  1862          MOVAB   UCB$L_TL_CTRLC(R5),R7   ; ASSUME CONTROL C
                05   11  0A4C  1863          BRB     CTRLAST
                         0A4E  1864 SET_CTRLY:
      57   0090 C5   9E  0A4E  1865          MOVAB   UCB$L_TL_CTRLY(R5),R7   ; ADDRESS LIST HEAD
                         0A53  1866 CTRLAST:
                52   D4  0A53  1867          CLRL    R2                      ; NULL MASK
        00000000'GF  16  0A55  1868          JSB     G^COM$SETATTNAST        ; ENTER SET UP CODE
   1A 20 A3     07   E1  0A5B  1869          BBC     #IO$V_CTRLYAST, -       ; BR IF NOT ENABLING CTRL-Y
                         0A60  1870                  IRP$W_FUNC(R3),10$      ; ASTS
                03   E5  0A60  1871          BBCC    #UCB$V_TT_HANGUP,-      ; CHECK FOR LOST HANGUP NOTIFICATION
           15 68 A5      0A62  1872                  UCB$W_DEVSTS(R5),10$
           54   57   D0  0A65  1873          MOVL    R7,R4                   ; DELIVER LOST HANGUP AST
           50   64   D0  0A68  1874          MOVL    (R4),R0                 ; GET AST BLOCK ADDRESS
           02CC 8F   3C  0A6B  1875          MOVZWL  #SS$_HANGUP,-           ; SIGNAL SPECIAL HANGUP STATUS
              1C A0      0A6F  1876                  ACB$L_KAST+4(R0)
        00000000'GF  16  0A71  1877          JSB     G^COM$DELATTNAST        ; AND FIRE THE AST
           50   01   3C  0A77  1878          MOVZWL  #SS$_NORMAL,R0
                         0A7A  1879
                         0A7A  1880 10$:
        00000000'GF  17  0A7A  1881          JMP     G^EXE$FINISHIOC
```

```
                              0A80
                              0A80   ; PROCESS CONNECT/DISCONNECT FUNCTIONS
                              0A80
                              0A80   SET_DISCONNECT:
    00000000'EF    28 A5   D1 0A80              CMPL    UCB$L_DDB(R5),VT$DDB        ; IS THIS TERMINAL  VIRTUAL TERMINAL
                      09   12 0A88              BNEQU   10$                         ; NO THEN DON'T LET THE USER LOGOUT
 20 A3    06    00    08   F0 0A8A              INSV    #TTY$C_FC_DISCON,#IRP$V_FCODE,#IRP$S_FCODE,IRP$W_FUNC(R3);
                    FF0D   31 0A90              BRW     QPKT                        ; QUEUE PACKET FOR FOLLOW ON PROCESSING
                              0A93
          50    0144 8F   3C 0A93   10$:       MOVZWL  #SS$_IVDEVNAM,R0            ; INVALID DEVICE NAME
            00000000'GF   17 0A98              JMP     G^EXE$ABORTIO               ; ABORT THE COMMAND
```

```
                       0A9E  1895 ;
                       0A9E  1896 ; Normal exit
                       0A9E  1897 ;
                       0A9E  1898 SET_DONE:
       50   01    D0   0A9E  1899          MOVL    #SS$_NORMAL,R0
00000000'GF       17   0AA1  1900          JMP     G^EXE$FINISHIOC
```

```
                          0AA7  1902
                          0AA7  1903          ; HANGUP FUNCTION
                          0AA7  1904
                          0AA7  1905 SET_HANGUP:
  20 A3   06   00   04  F0 0AA7  1906         INSV    #TTY$C_FC_HANGUP,#IRP$V_FCODE,#IRP$S_FCODE,IRP$W_FUNC(R3);
                  FEF0  31 0AAD  1907         BRW     QPKT                            ; HANGUP IGNORES P1-PN.
```

```
                                  0AB0  1909
                                  0AB0  1910            ; LOOPBACK/CONTROL  DIAGNOSTIC FUNCTIONS
                                  0AB0  1911
                                  0AB0  1912 SET_MAINT:
                        0A    E1  0AB0  1913         BBC     #IO$V_SET_MODEM,-           ; IF SET MODEM DIAGNOSTIC FUNCTION
                  03 20 A3        0AB2  1914                 IRP$W_FUNC(R3),12$          ; THEN OK
                      002D    30  0AB5  1915         BSBW    GET_PARAMS                  ; GET USER PARAMETERS
        20 A3    06    00   05    F0  0AB8  1916 12$:    INSV    #TTY$C_FC_MAINT,#IRP$V_FCODE,#IRP$S_FCODE,IRP$W_FUNC(R3);
                      FEDF    31  0ABE  1917         BRW     QPKT                        ; DONE
                                  0AC1  1918
```

```
                          0AC1  1920
                          0AC1  1921 ; ENABLE OUT OF BAND MASK
                          0AC1  1922
                          0AC1  1923 SET_OUTBAND:
     57   009C C5     9E  0AC1  1924          MOVAB   UCB$L_TL_BANDQUE(R5),R7          ; GET LIST HEAD ADDRESS
     52   0098 C5     DE  0AC6  1925          MOVAL   UCB$L_TL_OUTBAND(R5),R2          ; GET CURRENT MASK ADDRESS
      00000000'GF     16  0ACB  1926          JSB     G^COM$SETCTRLAST                 ; ENABLE/DISABLE AST
      00000000'GF     17  0AD1  1927          JMP     G^EXE$FINISHIOC                  ; DONE
```

```
          00000000'GF   17  0AD7  1929            JMP     G^EXE$FINISHIOC                        ; DONE
                            0ADD  1930
                            0ADD  1931 ; SET CONTROLLING PROCESS PID   (FOR ^C,^Y,OOB PURPOSES)
                            0ADD  1932
                            0ADD  1933 SET_PID:
      00A4 C5    60 A4  D0  0ADD  1934            MOVL    PCB$L_PID(R4),UCB$L_TL_CTLPID(R5)
                    B9  11  0AE3  1935            BRB     SET_DONE
```

```
                         0AE5  1937
                         0AE5  1938 ;
                         0AE5  1939 ;       PROBE USER BUFFER AND GET PARAMETERS
                         0AE5  1940 ;
                         0AE5  1941
                         0AE5  1942 GET_PARAMS:
          51    6C   D0  0AE5  1943         MOVL    P1(AP),R1                   ; GET ADDRESS OF BUFFER
          50    0C   3C  0AE8  1944         MOVZWL  #SS$_ACCVIO,R0              ; ASSUME ACCESS VIOLATION
                         0AEB  1945         IFNORD  #8,(R1),10$                 ; BR IF NO ACCESS TO QUADWORD BUFFER
       38 A3    61   D0  0AF1  1946         MOVL    (R1),IRP$L_MEDIA(R3)        ; SAVE NEW DATA
    40 A3    04 A1   D0  0AF5  1947         MOVL    4(R1),IRP$Q_TT_STATE(R3);   SAVE 1ST DEVDEPEND WORD
       32 A3    08   B0  0AFA  1948         MOVW    #8,IRP$W_BCNT(R3)           ; INDICATE DEFAULT SIZE
       0C    04 AC   D1  0AFE  1949         CMPL    P2(AP),#T2                  ; CHECK FOR SECOND DEVDEPEND ARGUMENT
                0F   1F  0B02  1950         BLSSU   5$                          ; NONE
                         0B04  1951         IFNORD  #12,(R1),10$                ; CHECK IF ADDRESSABLE
    44 A3    08 A1   D0  0B0A  1952         MOVL    8(R1),IRP$Q_TT_STATE+4(R3)
                         0B0F  1953                                             ; GET 2ND DEVDEPEND WORD
       32 A3    0C   B0  0B0F  1954         MOVW    #12,IRP$W_BCNT(R3)          ; LENGTH OF ADDITIONAL DATA
                     05  0B13  1955 5$:     RSB
   00000000'GF       17  0B14  1956 10$:    JMP     G^EXE$ABORTIO               ; RETURN ON ERROR
```

```
                             0B1A  1959          .SBTTL  MODE OR CHARACTERISTICS
                             0B1A  1960 ;++
                             0B1A  1961 ; TTY$FDTSENSEM - SENSE MODE
                             0B1A  1962 ; TTY$FDTSENSEC - SENSE CHARACTERISTICS
                             0B1A  1963 ;
                             0B1A  1964 ; FUNCTIONAL DESCRIPTION:
                             0B1A  1965 ;
                             0B1A  1966 ; THIS ROUTINE PASSES THE CURRENT CHARACTERISTICS FOR SENSEMODE AND
                             0B1A  1967 ; THE PERMANENT CHARACTERISTICS FOR SETCHAR.
                             0B1A  1968 ; THE BUFFER RETURNED IS A QUADWORD.
                             0B1A  1969 ;
                             0B1A  1970 ; INPUTS:
                             0B1A  1971 ;
                             0B1A  1972 ;        R3 = I/O PACKET ADDRESS
                             0B1A  1973 ;        R4 = CURRENT PCB ADDRESS
                             0B1A  1974 ;        R5 = UCB ADDRESS
                             0B1A  1975 ;        R6 = CCB ADDRESS
                             0B1A  1976 ;        R7 = FUNCTION CODE
                             0B1A  1977 ;        AP = ARG LIST FROM QIO
                             0B1A  1978 ;
                             0B1A  1979 ; OUTPUTS:
                             0B1A  1980 ;
                             0B1A  1981 ;        CONTROL IS PASSED TO EXE$ABORTIO ON FAILURE
                             0B1A  1982 ;        OR COMPLETED VIA EXE$FINISHIO
                             0B1A  1983 ;
                             0B1A  1984 ; STATUS RETURNS:
                             0B1A  1985 ;
                             0B1A  1986 ;        SS$_NORMAL - SUCCESSFULL
                             0B1A  1987 ;        SS$_ACCVIO - BUFFER NOT ACCESSIBLE
                             0B1A  1988 ;--
                             0B1A  1989 TTY$FDTSENSEM::                                  ; SENSE MODE
                0159   30    0B1A  1990         BSBW    VERIFY_SENSE                     ; VERIFY USER STORAGE
                       B3    0B1D  1991         BITW    #<IO$M_TYPEAHDCNT!-              ; TEST FOR SPECIAL MODIFIERS
                             0B1E  1992                 IO$M_RD_MODEM!IO$M_BRDCST>,-
    20 A3    40C0 8F         0B1E  1993                 IRP$W_FUNC(R3)
                  03   13    0B23  1994         BEQL    10$                              ; SKIP IF NOT
                0103   31    0B25  1995         BRW     GET_SPECIAL                      ; DO THEM
                             0B28  1996
          61    40 A5  D0    0B28  1997 10$:    MOVL    UCB$B_DEVCLASS(R5),(R1)  ; BUILD CLASS,TYPE, AND BUFFER SIZE
       04 A1    44 A5  D0    0B2C  1998         MOVL    UCB$L_DEVDEPEND(R5),4(R1);RETURN 1ST CHARACTERISTICS LONGWORD
             0C   50   91    0B31  1999         CMPB    R0,#12                           ; SECOND CHARACTERISTICS REQUESTED?
                  09   19    0B34  2000         BLSS    20$
                00B3   30    0B36  2001         BSBW    GET_DCL                          ; BUILD SPECIAL CHARACTERISTICS
  08 A1    48 A5  52   C9    0B39  2002         BISL3   R2,UCB$L_DEVDEPND2(R5),8(R1);AND 2ND LONGWORD (IF REQUESTED)
                  50   D4    0B3F  2003 20$:    CLRL    R0                               ; INIT RETURN
                  51   D4    0B41  2004         CLRL    R1
                             0B43  2005         DSBINT  UCB$B_FIPL(R5)                   ; INTERLOCK DISCONNECTS
       59    00A0 C5   D0    0B4A  2006         MOVL    UCB$L_TL_PHYUCB(R5),R9   ; GET PUCB ADDRESS
                  16   13    0B4F  2007         BEQL    30$                              ; DISCONNECTED FROM CHARS
       50    00F2 C9   D0    0B51  2008         MOVL    UCB$W_TT_SPEED-2(R9),R0  ; RETURN SPEED
       51    00F6 C9   D0    0B56  2009         MOVL    UCB$B_TT_PARITY-2(R9),R1;RETURN PARITY INFO
    51   FF000000 8F   CA    0B5B  2010         BICL    #^XFF000000,R1                   ; ZERO HIGH BYTE
       51    00F6 C9   B0    0B62  2011         MOVW    UCB$B_TT_CRFILL(R9),R1   ; AND CR/LF FILL
                             0B67  2012 30$:    ENBINT
                0097   31    0B6A  2013         BRW     GET_EXIT1                        ; EXIT RETURNING R1
                             0B6D  2014
                             0B6D  2015 ;               THIS ROUTINE PROCESSES SENSE CHARACTERISTICS FUNCTIONS
```

```
                        0B6D
                        0B6D  TTY$FDTSENSEC::                                        ; SENSE CHAR
         0106   30      0B6D            BSBW    VERIFY_SENSE                        ; VERIFY USER STORAGE
                B3      0B70            BITW    #<IO$M_TYPEAHDCNT!-                 ; TEST FOR SPECIALS
                        0B71                    IO$M_RD_MODEM!IO$M_BRDCST>,-
 20 A3   40C0 8F        0B71                    IRP$W_FUNC(R3)
           03   13      0B76            BEQL    10$                                 ; SKIP IF NOT
         00B0   31      0B78            BRW     GET_SPECIAL                         ; DO THEM
                        0B7B
                        0B7B  10$:
         006E   30      0B7B            BSBW    GET_DCL                             ; BUILD SPECIAL CHARACTERISTICS
      0DC0 8F   BB      0B7E            PUSHR   #^M<R6,R7,R8,R10,R11>
                        0B82            DSBINT  UCB$B_FIPL(R5)                      ; INTERLOCK DISCONNECTS
    59   00A0 C5   D0   0B89            MOVL    UCB$L_TL_PHYUCB(R5),R9              ; GET PUCB ADDRESS
           4A   13      0B8E            BEQL    30$                                 ; DISCONNECTED FROM CHARS
    56   00EF C9   D0   0B90            MOVL    UCB$B_TT_DETYPE-1(R9),R6; BUILD TYPE, AND BUFFER SIZE
     56   00'8F   90    0B95            MOVB    #DC$_TERM,R6                        ; BUILD DEVICE CLASS
    57   00C4 C9   D0   0B99            MOVL    UCB$L_TT_DECHAR(R9),R7              ;RETURN 1ST CHARACTERISTICS LONGWORD
 58   00C8 C9   52   C9 0B9E            BISL3   R2,UCB$L_TT_DECHA1(R9),R8;AND 2ND LONGWORD (IF REQUESTED)
                        0BA4  20$:
    5A   00E6 C9   D0   0BA4            MOVL    UCB$W_TT_DESPEE-2(R9),R10; RETURN SPEED
    5B   00EA C9   D0   0BA9            MOVL    UCB$B_TT_DEPARI-2(R9),R11; RETURN PARITY INFO
 5B   FF000000 8F  CA   0BAE            BICL    #^XFF000000,R11                     ; ZERO HIGH BYTE
    5B   00EA C9   B0   0BB5            MOVW    UCB$B_TT_DECRF(R9),R11              ; AND CR/LF FILL
                        0BBA            ENBINT                                      ; RELEASE INTERLOCK
           61   56   D0 0BBD            MOVL    R6,(R1)                             ; RETURN USER DATA
        04 A1   57   D0 0BC0            MOVL    R7,4(R1)
           0C   50   91 0BC4            CMPB    R0,#12                              ; DID HE ASK FOR 2ND ?
                04   19 0BC7            BLSS    25$                                 ; NO
        08 A1   58   D0 0BC9            MOVL    R8,8(R1)
           50   5A   D0 0BCD  25$:      MOVL    R10,R0                              ; RETURN IOSB DATA
           51   5B   D0 0BD0            MOVL    R11,R1
         0DC0 8F   BA   0BD3            POPR    #^M<R6,R7,R8,R10,R11>               ; RESTORE SCRATCH REGISTERS
            002A   31   0BD7            BRW     GET_EXIT1                           ; EXIT RETURNING R0,R1
                        0BDA
                        0BDA  30$:      ENBINT                                      ; RELEASE INTERLOCK
         0DC0 8F   BA   0BDD            POPR    #^M<R6,R7,R8,R10,R11>               ; RESTORE SCRATCH REGISTERS
            FF36   31   0BE1            BRW     TTY$FDTSENSEM                       ; AND TREAT AS SENSE MODE
```

```
                      0BE4   2055           .sbttl  SETMODE/CHAR service routines
                      0BE4   2056 GET_BRDCST:
61     00A8 C5    7D  0BE4   2057           MOVQ    UCB$Q_TL_BRKTHRU(R5),(R1)          ; RETURN BROADCAST MASK
            000F  31  0BE9   2058           BRW     GET_EXIT
```

```
                             0BEC  2060 ;          THIS ROUTINE BUILDS DCL PRIVATE CHARACTERISTICS
                             0BEC  2061
                             0BEC  2062 GET_DCL:
               52       D4   0BEC  2063            CLRL    R2                          ; INIT RETRUN ARGUMENT
            60 A5       D5   0BEE  2064            TSTL    UCB$L_AMB(R5)               ; ANY ASSOCIATED MAILBOX?
               07       13   0BF1  2065            BEQL    5$                          ; NO
   52  00000200 8F      C8   0BF3  2066            BISL    #TT2$M_DCL_MAILBX,R2        ; YES, SO BUILD CHARACTERISTIC
                        05   0BFA  2067 5$:        RSB
                        0BFB  2069
                        0BFB  2070 GET_EXIT:
      50    01    B0    0BFB  2071            MOVW    #SS$_NORMAL,R0
 00000000'GF      17    0BFE  2072            JMP     G^EXE$FINISHIOC          ; COMPLETE REQUEST IOSB WORD 0
                        0C04  2073 GET_EXIT1:
      50    01    B0    0C04  2074            MOVW    #SS$_NORMAL,R0
 00000000'GF      17    0C07  2075            JMP     G^EXE$FINISHIO           ; COMPLETE REQUEST IOSB WORD 0,1
```

```
                            0C0D  2077
                            0C0D  2078 ;            THIS ROUTINE BUILDS CONTROLLER TYPE AND RECEIVE MODEM SIGNALS
                            0C0D  2079
                            0C0D  2080 GET_MODEM:
           50    24 A9   D0 0C0D  2081          MOVL    UCB$L_CRB(R9),R0                ; GET CRB ADDRESS
           56    0B A0   90 0C11  2082          MOVB    CRB$B_TT_TYPE(R0),R6            ; RETURN CRB TYPE
        57     0124 C9   90 0C15  2083          MOVB    UCB$B_TT_DS_RCV(R9),R7          ; RETURN CURRENT RECEIVE MODEM SIGNALS
                            0C1A  2084          ENBINT                                  ; RELEASE INTERLOCK
           61       56   90 0C1D  2085          MOVB    R6,(R1)                         ; RETURN USER DATA
        02 A1       57   90 0C20  2086          MOVB    R7,2(R1)
           00C0 8F       BA 0C24  2087          POPR    #^M<R6,R7>                      ; RESTORE SCRATCH DATA
                FFD0     31 0C28  2088          BRW     GET_EXIT
```

```
                           0C2B  2090
                           0C2B  2091 ;            THIS ROUTINE PROCESSES MODIFIERS
                           0C2B  2092
                           0C2B  2093 GET_SPECIAL:
              0E     E0    0C2B  2094              BBS     #IO$V_BRDCST,-          ; ONLY REQUIRES LUCB
        B4 20 A3           0C2D  2095                      IRP$W_FUNC(R3),GET_BRDCST
         00C0 8F     BB    0C30  2096              PUSHR   #^M<R6,R7>              ; SAVE SCRATCH
                           0C34  2097              DSBINT  UCB$B_FIPL(R5)          ; INTERLOCK DISCONNECTS TO PCUB
    59   00A0 C5     D0    0C3B  2098              MOVL    UCB$L_TL_PHYUCB(R5),R9  ; GET PUCB ADDRESS
              08     13    0C40  2099              BEQL    10$                     ; DISCONNECTED (RETURN ZERO)
  0D 20 A3    06     E0    0C42  2100              BBS     #IO$V_TYPEAHDCNT,IRP$W_FUNC(R3),GET_TYPEAHD
            FFC3     31    0C47  2101              BRW     GET_MODEM
                           0C4A  2102 10$:         ENBINT                          ; RELEASE INTERLOCK
         00C0 8F     BA    0C4D  2103              POPR    #^M<R6,R7>
            FFA7     31    0C51  2104              BRW     GET_EXIT
```

```
                          0C54  2106
                          0C54  2107 ;       THIS ROUTINE BUILDS THE NUMBER OF CHARACTERS IN TYPEAHD
                          0C54  2108 ;       AND RETURNS THE TOP CHARACTER IN THE BUFFER
                          0C54  2109
                          0C54  2110 GET_TYPEAHD:
                 56   7C  0C54  2111         CLRQ    R6                          ; INIT RETURN VALUES (R6,R7)
     50    00E4 C9    D0  0C56  2112         MOVL    UCB$L_TT_TYPAHD(R9),R0      ; ADDRESS OF TYPEAHD BUFFER
                 08   13  0C5B  2113         BEQL    10$                         ; SKIP IF DISCONNECTED
      56     0C A0    B0  0C5D  2114         MOVW    TTY$W_TA_INAHD(R0),R6       ; GET NUMBER CHARACTERS IN BUFFER
      57     04 B0    90  0C61  2115         MOVB    @TTY$[_TA_GET(R0),R7        ; GET LOOKAHEAD CHARACETER
                          0C65  2116 10$:    ENBINT                              ; RELEASE INTERLOCK
            61   56   B0  0C68  2117         MOVW    R6,(R1)                     ; RETURN USER DATA
         02 A1   57   90  0C6B  2118         MOVB    R7,2(R1)
          00C0 8F     BA  0C6F  2119         POPR    #^M<R6,R7>                  ; RESTORE SCRATCH DATA
             FF85     31  0C73  2120         BRW     GET_EXIT
```

```
                          0C76
                          0C76   ;          THIS ROUTINE VERIFIES THAT THE USER BUFFER IS ACCESSABLE
                          0C76
                          0C76   VERIFY_SENSE:                                                ;
        51    6C    D0    0C76              MOVL      P1(AP),R1                             ; ADDRESS USER BUFFER
                          0C79              IFNOWRT   #8,(R1),30$                           ; BR IF NO ACCESS TO QUADWORD BUFFER
        50    08    D0    0C7F              MOVL      #8,R0                                 ; INIT DEFAULT ARGUMENT SIZE
              61    7C    0C82              CLRQ      (R1)                                  ; INIT RETURN DATA
   52    04   AC    D0    0C84              MOVL      P2(AP),R2                             ; GET SIZE ARGUMENT
      0C      52    D1    0C88              CMPL      R2,#12                                ; ROOM FOR SECOND DEVDEPEND SPECIFIED?
              0C    1F    0C8B              BLSSU     25$                                   ; NO
                          0C8D              IFNOWRT   #12,(R1),30$                          ; CHECK IF WRITE ACCESS
        50    0C    B0    0C93              MOVW      #12,R0                                ; SAVE ARGUMENT SIZE
           08 A1    D4    0C96              CLRL      8(R1)                                 ; INIT RETURN FIELD
                          0C99   25$:
                    05    0C99              RSB
                          0C9A   30$:
        50    0C    3C    0C9A              MOVZWL    #SS$_ACCVIO,R0                        ; SET ERROR STATUS
     00000000'GF    17    0C9D              JMP       G^EXE$ABORTIO                         ; ABORT THE IO
```

```
         0CA3  2142
```

```
                    0CA3  2144          .SBTTL  MOVE TRANSLATE - TRANSLATE TO UPPERCASE
                    0CA3  2145  ;++
                    0CA3  2146  ;
                    0CA3  2147  ; MOVE_TRANSLATE
                    0CA3  2148  ;
                    0CA3  2149  ; THIS ROUTINE MOVES AND TRANSLATES LOWERCASE OUTPUT DATA FOR
                    0CA3  2150  ; UPPERCASE TERMINALS and TRANSLATES 8-BIT CHARACTERS TO 7-BIT FALLBACK
                    0CA3  2151  ; PRESENTATION FOR TERMINALS THAT DO NOT UNDERSTAND THE 8 BIT REPRESENTATIONS.
                    0CA3  2152  ; IT PARSES ESCAPE SEQUENCES FOR ANSI, VT100
                    0CA3  2153  ; AND VT52 TERMINALS, AND REFRAINS FROM TRANSLATION OF ANY DATA
                    0CA3  2154  ; IN ESCAPE OR CONTROL SEQUENCES.
                    0CA3  2155  ;
                    0CA3  2156  ; INPUTS:
                    0CA3  2157  ;               R2 = DESTINATION ADDRESS
                    0CA3  2158  ;               R3 = destination length
                    0CA3  2159  ;               R5 = UCB ADDRESS
                    0CA3  2160  ;               R6 = SOURCE ADDRESS
                    0CA3  2161  ;               R7 = LENGTH
                    0CA3  2162  ;               R8 = LOW BIT SET MEANS DO LOWERCASEING
                    0CA3  2163  ;
                    0CA3  2164  ; OUTPUTS:
                    0CA3  2165  ;               R2 = END OF DESTINATION STRING +1
                    0CA3  2166  ;               R5 = UCB ADDRESS
                    0CA3  2167  ;
                    0CA3  2168  ;               R0 - R4 DESTROYED
                    0CA3  2169
                    0CA3  2170 MOVE_TRANSLATE:
                    0CA3  2171
         57    D5   0CA3  2172          TSTL    R7                      ; NULL STRING
         0B    13   0CA5  2173          BEQL    30$                     ; YES
      09 58    E8   0CA7  2174          BLBS    R8,5$                   ; ARE WE DOING LOWER?? YES THEN DO FALLBACK
         55    DD   0CAA  2175          PUSHL   R5                      ; ELSE JUST DO FALLBACK
       0064    30   0CAC  2176          BSBW    TTY$FALLBACK            ; CALL FALLBACK
         55  8ED0   0CAF  2177          POPL    R5                      ; AND RESTORE THE UCB ADDRESS
               05   0CB2  2178 30$:     RSB
                    0CB3  2179
         0E    E1   0CB3  2180 5$:      BBC     #TT2$V_FALLBACK,-
   0D 48 A5         0CB5  2181                  UCB$L_DEVDEPND2(R5),TTY$UPPER; NO FALLBACK THEN ONLY UPPER CASE
         2C    BB   0CB8  2182          PUSHR   #^M<R2,R3,R5>           ; SAVE THE DESTROYED REGISTERS
       0056    30   0CBA  2183          BSBW    TTY$FALLBACK            ; CALL FALLBACK
         2C    BA   0CBD  2184          POPR    #^M<R2,R3,R5>           ; THEN RETURN THE DATA
    57   53    D0   0CBF  2185          MOVL    R3,R7                   ; UPDATE THE LENGTH AND ADDRESS
    56   52    D0   0CC2  2186          MOVL    R2,R6                   ; TO TRANSLATE TO UPPER
```

```
                          0CC5  2188          .SBTTL  TTY$UPPER - Translate a string to upper case
                          0CC5  2189 ;++
                          0CC5  2190 ;TTY$UPPER - Upper case translation
                          0CC5  2191 ;
                          0CC5  2192 ; Description:
                          0CC5  2193 ;
                          0CC5  2194 ;          Given an input string it will take all of the lower case characters
                          0CC5  2195 ; in it and change it to upper case (characters in escape sequences are not
                          0CC5  2196 ; bothered).
                          0CC5  2197 ;
                          0CC5  2198 ; Inputs:
                          0CC5  2199 ;          R2 = DESTINATION ADDRESS
                          0CC5  2200 ;          R5 = UCB ADDRESS
                          0CC5  2201 ;          R6 = SOURCE ADDRESS
                          0CC5  2202 ;          R7 = LENGTH
                          0CC5  2203 ; Outputs:
                          0CC5  2204 ;
                          0CC5  2205 ;          R2 = END OF DESTINATION STRING +1
                          0CC5  2206 ;          R5 = UCB ADDRESS
                          0CC5  2207 ;
                          0CC5  2208 ;          R0 - R4 DESTROYED
                          0CC5  2209 ;--
                          0CC5  2210 TTY$UPPER:
          53    86    9A  0CC5  2211          MOVZBL  (R6)+,R3                    ; GET NEXT CHAR TO MOVE
 04     0000'CF43     91  0CC8  2212          CMPB    W^TTY$A_CCLIST[R3],#TTY$K_ET_ESCAPE; CHECK OUT ESCAPE SEQUENCES
                11    13  0CCE  2213          BEQL    40$
                03    E1  0CD0  2214          BBC     #TTY$V_CH_LOWER,-           ; SKIP IF NOT LOWER CASE
   03 0000'CF43           0CD2  2215                  W^TTY$A_TYPE[R3],20$
          53    20    AA  0CD7  2216          BICW    #^X20,R3                    ; CONVERT TO UPPER CASE
                          0CDA  2217 20$:
          82    53    90  0CDA  2218          MOVB    R3,(R2)+
             E5 57    F5  0CDD  2219          SOBGTR  R7,TTY$UPPER                            ; CONTINUE UNTIL DONE
                          0CE0  2220 30$:
                      05  0CE0  2221          RSB
                          0CE1  2222
                          0CE1  2223 40$:
          82    53    90  0CE1  2224          MOVB    R3,(R2)+                    ; COPY ESCAPE
                57    D7  0CE4  2225          DECL    R7                          ; ADJUST COUNT
                F8    13  0CE6  2226          BEQL    30$                         ; QUIT IF DONE
                18    E0  0CE8  2227          BBS     #TT2$V_ANSICRT,-            ; CHECK FOR DEVICES WITH ANSI SEQUENCES
          0C 48 A5        0CEA  2228                  UCB$L_DEVDEPND2(R5),45$
                03    E0  0CED  2229          BBS     #TT$V_ESCAPE,-
          07 44 A5        0CEF  2230                  UCB$L_DEVDEPEND(R5),45$
   41 A5     40 8F    91  0CF2  2231          CMPB    #TT$_VT5X,UCB$B_DEVTYPE(R5)
                CC    1A  0CF7  2232          BGTRU   TTY$UPPER
                          0CF9  2233 ;
                          0CF9  2234 45$:
                51    DD  0CF9  2235          PUSHL   R1                          ; SAVE R1
             F302'    30  0CFB  2236          BSBW    ESCINIT                     ; INIT THE ESCAPE SEQUENCE RULES
          54    51    9A  0CFE  2237          MOVZBL  R1,R4                       ;...
             51 8ED0      0D01  2238          POPL    R1                          ; RESTORE R1
                          0D04  2239 50$:
          53    86    9A  0D04  2240          MOVZBL  (R6)+,R3                    ; GET NEXT SEQUENCE CHARACTER
             F2F6'    30  0D07  2241          BSBW    E_SYNTAX                    ; CHECK ESCAPE SEQUENCE SYNTAX
                CE    15  0D0A  2242          BLEQ    20$                         ; ENDED OK, OR FAILURE
          82    53    90  0D0C  2243          MOVB    R3,(R2)+
             F2 57    F5  0D0F  2244          SOBGTR  R7,50$                      ; CONTINUE TILL END
```

```
                                     05  0D12  2245        RSB
```

```
                                   0D13          .SBTTL  TTY$FALLBACK - ROUTINE TO TRANSLATE 8-BIT CHARACTERS TO 7-BIT
                                   0D13     ;++
                                   0D13     ; TTY$FALLBACK - SUBROUTINE THAT WILL MOVE A STRING AND INSERT FALLBACK REPRESENTATI
                                   0D13     ;
                                   0D13     ; DESCRIPTION:
                                   0D13     ;
                                   0D13     ;         This routine will take an input buffer of any length and take any
                                   0D13     ; eight bit characters in it and change them into their 7-bit fallback presentation
                                   0D13     ; This includes inserting characters for multi-character expansions.
                                   0D13     ;
                                   0D13     ; Inputs:
                                   0D13     ;         R2 - destination address
                                   0D13     ;         R3 - destination length
                                   0D13     ;         R5 - ucb address
                                   0D13     ;         R6 - Source address
                                   0D13     ;         R7 - Source length
                                   0D13     ;
                                   0D13     ; Outputs:
                                   0D13     ;         R2 - End of destination string
                                   0D13     ;         R0 - R5 Destroied
                                   0D13     ;
                                   0D13     ; MOVE FALLBACK TRANSLATED
                                   0D13     ;
                                   0D13     TTY$FALLBACK:
                     50   57   D0  0D13             MOVL    R7,R0               ; load up the necessary registers
                     51   56   D0  0D16             MOVL    R6,R1
                     54   53   D0  0D19             MOVL    R3,R4
                     55   52   D0  0D1C             MOVL    R2,R5
                                   0D1F     310$:
 00000000'FF    FF 8F   61   50   2F  0D1F             MOVTUC  R0,(R1),#255,@TTY$A_FALLTAB,R4,(R5); TRANSLATE WHAT WE CAN
                     65   54       0D29
                          04   1D  0D2B             BVS     320$                ; IF WE CAN'T TRANSLATE IT THEN IT MUST BE LONG
                     52   55   D0  0D2D             MOVL    R5,R2               ; GET THE CURRENT END OF DATA
                               05  0D30             RSB                         ; RETURN
                                   0D31
                                   0D31     320$:
                     53   81   9A  0D31             MOVZBL  (R1)+,R3            ; GET THE CHARACTER
                          54   DD  0D34             PUSHL   R4                  ; SAVE A REGISTER
            54    00000000'EF   D0  0D36             MOVL    TTY$A_EXPTAB,R4     ; GET TEH TABLE ADDRESS
              53    FF6A C443   9A  0D3D             MOVZBL  -150(R4)[R3],R3     ; GET THE OFFSET TO THE STRING
                        54 8ED0    0D43             POPL    R4                  ; RESTORE OUR SCRATCH REGISTER
          52    00000000'FF43   9A  0D46             MOVZBL  @TTY$A_EXPAN[R3],R2; GET THE LENGTH OF THE SEQUENCE
                          53   D6  0D4E             INCL    R3                  ; MOVE BY THE COUNT
                          50   D7  0D50             DECL    R0                  ;...
          85    00000000'FF43   90  0D52     330$:   MOVB    @TTY$A_EXPAN[R3],(R5)+; MOVE IN THIS CHARACTER
                          54   D7  0D5A             DECL    R4                  ; TAKE OUT THE CHARACTER SLOT
                          C1   13  0D5C             BEQL    310$
                          53   D6  0D5E             INCL    R3                  ; MOVE OVER A CHARACTER
                       EF 52   F5  0D60             SOBGTR  R2,330$
                        FFB9   31  0D63             BRW     310$
                                   0D66     ;
                                   0D66     ; CALCULATE THE NUMBER OF CHARACTERS
                                   0D66     ; THAT WILL BE OUTPUT IN FALLBACK
                                   0D66     ;
                                   0D66     ADDFALL:
                          59   D4  0D66             CLRL    R9                  ; CLEAN OUT R9
 80 8F    00000000'FF   61   50   2A  0D68     210$:   SCANC   R0,(R1),@TTY$A_FALLTAB,#^X80; SCANN FOR FALLBACK CHARACTRS
```

```
                 01   12  0D72  2303            BNEQ    220$              ; NOT DONE THEN COUNT THE EXPANSION
                      05  0D74  2304            RSB
                          0D75  2305
                 54   DD  0D75  2306 220$:      PUSHL   R4                ; SAVE A REGISTER
              53 81   9A  0D77  2307            MOVZBL  (R1)+,R3          ; GET THE CHARACTER
      54   00000000'EF    D0  0D7A  2308        MOVL    TTY$A_EXPTAB,R4   ; GET THE TABLE ADDRESS
        53   FF6A C443    9A  0D81  2309        MOVZBL  -150(R4)[R3],R3   ; GET THE OFFSET TO THE STRING
               54 8ED0    0D87  2310            POPL    R4                ; RESTORE THE REGISTER
  52    00000000'FF43 9A  0D8A  2311            MOVZBL  @TTY$A_EXPAN[R3],R2; GET THE LENGTH OF THE SEQUENCE
                 52   D7  0D92  2312            DECL    R2                ; SUBTRACT 1 FROM THE LENGTH
                 50   D7  0D94  2313            DECL    R0                ;UPDATE THE SCAN POINTER
           59    52   C0  0D96  2314            ADDL    R2,R9
                 CD   11  0D99  2315            BRB     210$              ; AND CONTINUE COUNTING
                          0D9B  2316
                          0D9B  2317
                          0D9B  2318            .END
```

```
ACB$L_KAST          = 00000018               IO$V_DSABLMBX       = 0000000A
ADDFALL               00000D66 R       02    IO$V_ESCAPE         = 0000000E
ALTECHSTR             000005EE R       02    IO$V_EXTEND         = 0000000F
BAD_SET               00000967 R       02    IO$V_MAINT          = 00000006
BDPRMERR              000005DF R       02    IO$V_NOECHO         = 00000006
COM$DELATTNAST        ********    X    02    IO$V_NOFILTR        = 00000009
COM$SETATTNAST        ********    X    02    IO$V_NOFORMAT       = 00000008
COM$SETCTRLAST        ********    X    02    IO$V_PURGE          = 0000000B
CRB$B_TT_TYPE       = 0000000B               IO$V_REFRESH        = 0000000D
CTRLAST               00000A53 R       02    IO$V_SET_MODEM      = 0000000A
DC$_TERM              ********    X    02    IO$V_TIMED          = 00000007
DYN$C_TWP           = 00000030               IO$V_TRMNOECHO      = 0000000C
EDITMODE              00000617 R       02    IO$V_TYPEAHDCNT     = 00000006
ESCINIT               ********    X    02    IO$_READPBLK        = 0000000C
EXE$ABORTIO           ********    X    02    IO$_READPROMPT      = 00000037
EXE$ALLOCBUF          ********    X    02    IO$_TTYREADALL      = 0000003A
EXE$ALTQUEPKT         ********    X    02    IO$_TTYREADPALL     = 0000003B
EXE$BUFFRQUOTA        ********    X    02    IO$_WRITEPBLK       = 0000000B
EXE$CARRIAGE          ********    X    02    IOC$SEARCHDEV         ********    X    02
EXE$FINISHIO          ********    X    02    IPL$_SYNCH          = 00000008
EXE$FINISHIOC         ********    X    02    IRP$B_CARCON        = 0000003C
EXE$MAXACMODE         ********    X    02    IRP$L_MEDIA         = 00000038
EXE$PROBER            ********    X    02    IRP$L_SVAPTE        = 0000002C
EXE$QIODRVPKT         ********    X    02    IRP$L_TT_TERM       = 0000003C
EXE$QIORETURN         ********    X    02    IRP$L_VAL5          = 0000009C
EXE$READCHK           ********    X    02    IRP$M_TERMIO        = 00000200
EXE$WRITECHK          ********    X    02    IRP$Q_TT_STATE      = 00000040
E_SYNTAX              ********    X    02    IRP$S_FCODE         = 00000006
FILLCHR               00000641 R       02    IRP$V_FCODE         = 00000000
GET_BRDCST            00000BE4 R       02    IRP$V_FUNC          = 00000001
GET_DCL               00000BEC R       02    IRP$W_BCNT          = 00000032
GET_EXIT              00000BFB R       02    IRP$W_BOFF          = 00000030
GET_EXIT1             00000C04 R       02    IRP$W_FUNC          = 00000020
GET_LUCB              000009BD R       02    IRP$W_STS           = 0000002A
GET_MODEM             00000C0D R       02    IRP$W_TT_PRMPT      = 0000004C
GET_PARAMS            00000AE5 R       02    ITEMLOOP              00000343 R       02
GET_SPECIAL           00000C2B R       02    ITMREADERR            000005E6 R       02
GET_TYPEAHD           00000C54 R       02    JIB$L_BYTCNT        = 00000020
GOBAD                 000006B8 R       02    MODIFIERS             000006BB R       02
GOOD_SET              00000970 R       02    MOVE_TRANSLATE        00000CA3 R       02
INIOFFSET             00000658 R       02    P1                  = 00000000
INISTRNG              0000066B R       02    P2                  = 00000004
IO$M_BRDCST         = 00004000               P3                  = 00000008
IO$M_CVTLOW         = 00000100               P4                  = 0000000C
IO$M_DSABLMBX       = 00000400               P5                  = 00000010
IO$M_ESCAPE         = 00004000               P6                  = 00000014
IO$M_NOECHO         = 00000040               PCB$L_JIB           = 00000080
IO$M_NOFILTR        = 00000200               PCB$L_PID           = 00000060
IO$M_PURGE          = 00000800               PCB$L_UIC           = 000000BC
IO$M_RD_MODEM       = 00000080               PICSTRNG              000006F4 R       02
IO$M_REFRESH        = 00002000               PMS$GB_PROMPT         ********    X    02
IO$M_TIMED          = 00000080               PR$_IPL               ********    X    02
IO$M_TRMNOECHO      = 00001000               PROMPT                0000071D R       02
IO$M_TYPEAHDCNT     = 00000040               QPKT                  000009A0 R       02
IO$V_BRDCST         = 0000000E               SCH$GL_PCBVEC         ********    X    02
IO$V_CTRLYAST       = 00000007               SCH$IOLOCKW           ********    X    02
IO$V_CVTLOW         = 00000008               SCH$IOUNLOCK          ********    X    02
```

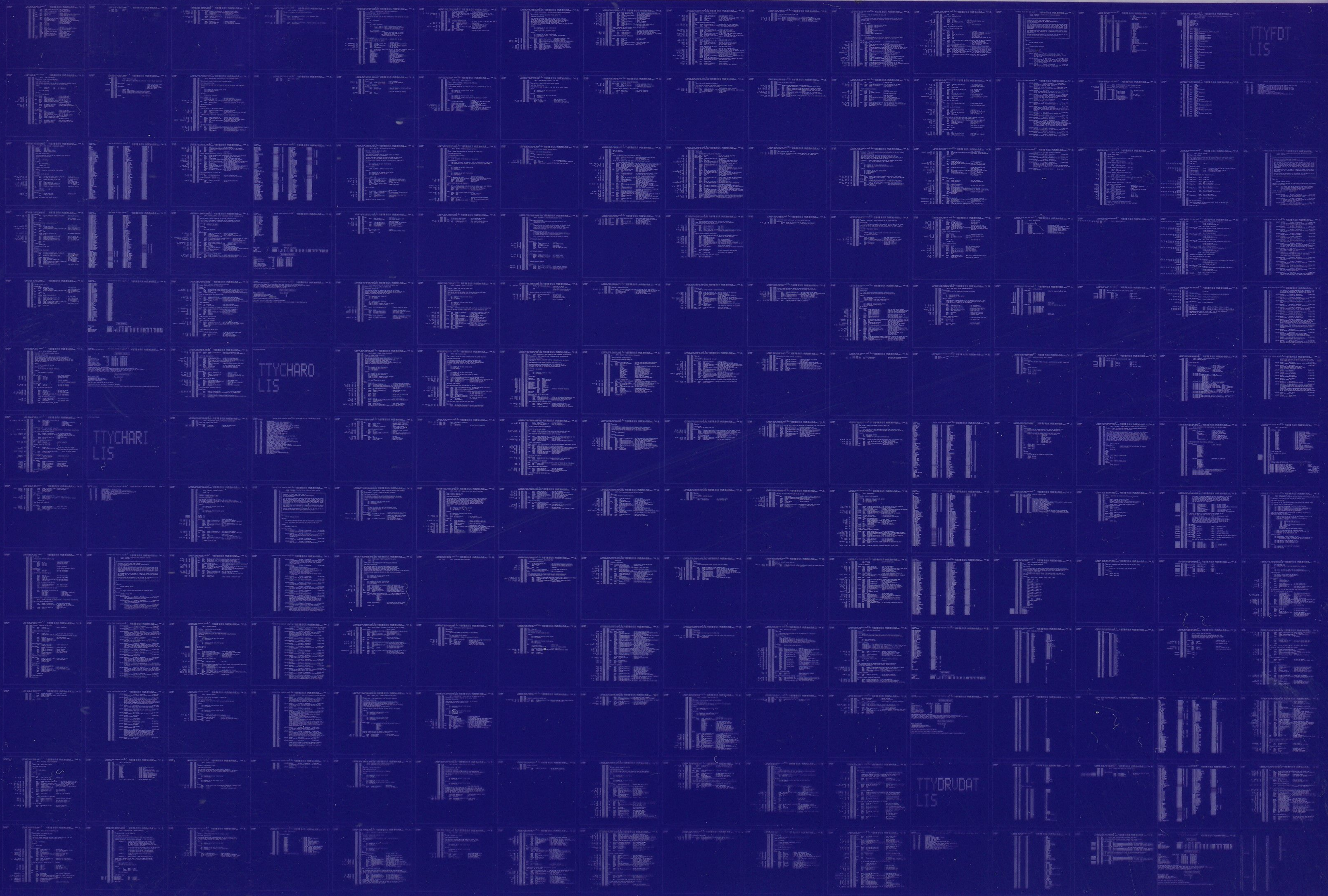
```
SET                           00000953 R      02  TT$_VT5X                  = 00000040
SET_BRDCST                    000009A6 R      02  TT2$M_DCL_MAILBX          = 00000200
SET_COMMON                    00000937 R      02  TT2$V_ANSICRT             = 00000018
SET_CONNECT                   000009B1 R      02  TT2$V_EDITING             = 0000000C
SET_CTRLC                     00000A47 R      02  TT2$V_FALLBACK            = 0000000E
SET_CTRLY                     00000A4E R      02  TT2$V_INSERT              = 0000000D
SET_DISCONNECT                00000A80 R      02  TT2$V_PASTHRU             = 00000012
SET_DONE                      00000A9E R      02  TTY$A_CCLIST                ********    X   02
SET_HANGUP                    00000AA7 R      02  TTY$A_EXPAN                 ********    X   02
SET_MAINT                     00000AB0 R      02  TTY$A_EXPTAB                ********    X   02
SET_OUTBAND                   00000AC1 R      02  TTY$A_FALLTAB               ********    X   02
SET_PID                       00000ADD R      02  TTY$A_RB_PRM              = 0000004A
SS$_ACCVIO                  = 0000000C            TTY$A_STANDARD              ********    X   02
SS$_BADPARAM                = 00000014            TTY$A_TYPE                  ********    X   02
SS$_DEVALLOC                = 00000840            TTY$B_RB_ECHLEN           = 0000000B
SS$_HANGUP                  = 000002CC            TTY$B_RB_RVFCLR           = 00000046
SS$_IVDEVNAM                = 00000144            TTY$B_RB_RVFFIL           = 00000047
SS$_NOPRIV                  = 00000024            TTY$B_WB_TYPE             = 0000000A
SS$_NORMAL                  = 00000001            TTY$C_CR                  = 0000000D
TERM                          0000076F R      02  TTY$C_FC_CONNECT          = 00000007
TIMEOUT                       000007A3 R      02  TTY$C_FC_DISCON           = 00000008
TRM$K_EM_RDVERIFY           = 00000001            TTY$C_FC_HANGUP           = 00000004
TRM$M_TM_AUTO_TAB           = 00040000            TTY$C_FC_MAINT            = 00000005
TRM$M_TM_CVTLOW             = 00000100            TTY$C_FC_READ             = 00000000
TRM$M_TM_DSABLMBX           = 00000400            TTY$C_FC_SETC             = 00000003
TRM$M_TM_ESCAPE             = 00004000            TTY$C_FC_SETM             = 00000002
TRM$M_TM_NOECHO             = 00000040            TTY$C_FC_WRITE            = 00000001
TRM$M_TM_NOEDIT             = 00008000            TTY$C_LF                  = 0000000A
TRM$M_TM_NOFILTR            = 00000200            TTY$C_MAXPAGWID           = 000001FF
TRM$M_TM_NORECALL           = 00010000            TTY$FALLBACK                00000D13 R      02
TRM$M_TM_PURGE              = 00000800            TTY$FDTITEMREAD             0000026A R      02
TRM$M_TM_REFRESH            = 00002000            TTY$FDTREAD                 00000000 RG     02
TRM$M_TM_R_JUST             = 00020000            TTY$FDTSENSEC               00000B6D RG     02
TRM$M_TM_TIMED              = 00000080            TTY$FDTSENSEM               00000B1A RG     02
TRM$M_TM_TRMNOECHO          = 00001000            TTY$FDTSETC                 00000931 RG     02
TRM$V_TM_CVTLOW             = 00000008            TTY$FDTSETM                 00000924 RG     02
TRM$V_TM_DSABLMBX           = 0000000A            TTY$FDTWRITE                000007BC RG     02
TRM$V_TM_ESCAPE             = 0000000E            TTY$K_ER_ECHLINE          = 00000002
TRM$V_TM_NOECHO             = 00000006            TTY$K_ER_RVECHO           = 0000000A
TRM$V_TM_NOEDIT             = 0000000F            TTY$K_ET_ESCAPE           = 00000004
TRM$V_TM_NOFILTR            = 00000009            TTY$K_IL_LENGTH           = 0000000C
TRM$V_TM_PURGE              = 0000000B            TTY$K_IS_LENGTH           = 00000058
TRM$V_TM_REFRESH            = 0000000D            TTY$L_IL_ADR              = 00000004
TRM$V_TM_R_JUST             = 00000011            TTY$L_IL_RETADR           = 00000008
TRM$V_TM_TIMED              = 00000007            TTY$L_IS_ACMODE           = 00000000
TRM$V_TM_TRMNOECHO          = 0000000C            TTY$L_IS_AES              = 00000048
TRM$_ALTECHSTR              = 00000009            TTY$L_IS_AESLEN           = 0000004C
TRM$_INIOFFSET              = 00000008            TTY$L_IS_BUF              = 00000008
TRM$_INISTRNG               = 00000005            TTY$L_IS_BUFLEN           = 0000000C
TRM$_LASTITM                = 0000000A            TTY$L_IS_EDITMODE         = 00000004
TRM$_PICSTRNG               = 00000006            TTY$L_IS_INI              = 00000010
TRM$_TERM                   = 00000003            TTY$L_IS_INIBUF           = 00000018
TT$V_ESCAPE                 = 00000003            TTY$L_IS_INILEN           = 00000014
TT$V_HALFDUP                = 00000014            TTY$L_IS_ITMLST           = 0000001C
TT$V_LOWER                  = 00000007            TTY$L_IS_LASTITM          = 00000020
TT$V_PASSALL                = 00000000            TTY$L_IS_MODIFY           = 00000024
TT$V_SCRIPT                 = 00000006            TTY$L_IS_PIC              = 00000028
```

```
TTY$L_IS_PICLEN               = 0000002C          TTY$W_RB_TXTOFF               = 0000003C
TTY$L_IS_PRM                  = 00000030          TTY$W_RB_TXTSIZ               = 00000040
TTY$L_IS_PRMBUF               = 00000038          TTY$W_TA_INAHD                = 0000000C
TTY$L_IS_PRMLEN               = 00000034          TTY$W_WB_BCNT                 = 0000002A
TTY$L_IS_SPECIFYED            = 0000003C          UCB$B_DEVCLASS                = 00000040
TTY$L_IS_TERM                 = 00000040          UCB$B_DEVTYPE                 = 00000041
TTY$L_IS_TERMLEN              = 00000044          UCB$B_FIPL                    = 0000000B
TTY$L_IS_TIMEOUT              = 00000050          UCB$B_TT_CRFILL               = 000000F6
TTY$L_RB_AES                  = 00000024          UCB$B_TT_DECRF                = 000000EA
TTY$L_RB_DATA                 = 0000004A          UCB$B_TT_DEPARI               = 000000EC
TTY$L_RB_ECHSTR               = 00000014          UCB$B_TT_DETYPE               = 000000F0
TTY$L_RB_LIN                  = 0000002C          UCB$B_TT_DS_RCV               = 00000124
TTY$L_RB_MOD                  = 00000020          UCB$B_TT_PARITY               = 000000F8
TTY$L_RB_PIC                  = 00000018          UCB$L_AMB                     = 00000060
TTY$L_RB_TERM                 = 0000001C          UCB$L_CRB                     = 00000024
TTY$L_RB_TXT                  = 00000000          UCB$L_DDB                     = 00000028
TTY$L_RB_UVA                  = 00000004          UCB$L_DEVDEPEND               = 00000044
TTY$L_TA_GET                  = 00000004          UCB$L_DEVDEPND2               = 00000048
TTY$L_WB_DATA                 = 00000030          UCB$L_PDT                     = 00000084
TTY$L_WB_END                  = 00000020          UCB$L_PID                     = 0000002C
TTY$L_WB_IRP                  = 00000024          UCB$L_TL_BANDQUE              = 0000009C
TTY$L_WB_NEXT                 = 0000001C          UCB$L_TL_CTLPID               = 000000A4
TTY$M_ST_ECHAES               = 02000000          UCB$L_TL_CTRLC                = 00000094
TTY$M_ST_EDITING              = 00100000          UCB$L_TL_CTRLY                = 00000090
TTY$M_ST_EDITREAD             = 00000200          UCB$L_TL_OUTBAND              = 00000098
TTY$M_ST_ESCAPE               = 00000800          UCB$L_TL_PHYUCB               = 000000A0
TTY$M_ST_NOECHO               = 00000008          UCB$L_TT_DECHA1               = 000000C8
TTY$M_ST_NOFLTR               = 00000040          UCB$L_TT_DECHAR               = 000000C4
TTY$M_ST_OVERSTRIKE           = 00800000          UCB$L_TT_TYPAHD               = 000000E4
TTY$M_ST_PROMPT               = 00000020          UCB$Q_TL_BRKTHRU              = 000000A8
TTY$M_ST_RDVERIFY             = 00000400          UCB$V_TT_HANGUP               = 00000003
TTY$M_ST_READ                 = 00001000          UCB$W_DEVSTS                  = 00000068
TTY$M_ST_REFRSH               = 00000400          UCB$W_TT_DESPEE               = 000000E8
TTY$M_ST_TERMNORM             = 01000000          UCB$W_TT_SPEED                = 000000F4
TTY$M_ST_WRITE                = 00000080          VERIFY_SENSE                    00000C76 R      02
TTY$UPPER                       00000CC5 R      02  VT$DDB                          ********   X    02
TTY$V_CH_LOWER                = 00000003
TTY$V_ST_EOL                  = 00000008
TTY$V_ST_NOECHO               = 00000003
TTY$V_ST_PASALL               = 00000002
TTY$V_ST_PROMPT               = 00000005
TTY$V_ST_REFRSH               = 0000000A
TTY$V_ST_WRTALL               = 00000004
TTY$W_IL_LEN                  = 00000000
TTY$W_IL_TYPE                 = 00000002
TTY$W_IS_FILLCHR              = 00000054
TTY$W_IS_INIOFF               = 00000056
TTY$W_RB_AESLEN               = 00000028
TTY$W_RB_CPZCUR               = 00000038
TTY$W_RB_LINOFF               = 00000030
TTY$W_RB_LINREST              = 00000032
TTY$W_RB_MODE                 = 00000044
TTY$W_RB_PICLEN               = 0000003E
TTY$W_RB_PRMLEN               = 00000034
TTY$W_RB_RDSTATE              = 0000002A
TTY$W_RB_SIZE                 = 00000008
TTY$W_RB_TIMOS                = 00000036
```

```
                                                  +----------------+
                                                  ! Psect synopsis !
                                                  +----------------+

PSECT name                        Allocation          PSECT No.  Attributes
----------                        ----------          ---------  ----------
.  ABS  .                         00000000  (     0.)  00 (  0.)  NOPIC   USR   CON   ABS   LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
$ABS$                             00000000  (     0.)  01 (  1.)  NOPIC   USR   CON   ABS   LCL NOSHR   EXE   RD     WRT NOVEC BYTE
$$$115_DRIVER                     00000D9B  (  3483.)  02 (  2.)  NOPIC   USR   CON   REL   LCL NOSHR   EXE   RD     WRT NOVEC LONG

                                               +------------------------+
                                               ! Performance indicators !
                                               +------------------------+

Phase                          Page faults    CPU Time       Elapsed Time
-----                          -----------    --------       ------------
Initialization                          33    00:00:00.04    00:00:00.52
Command processing                     118    00:00:00.36    00:00:02.73
Pass 1                                 668    00:00:19.69    00:01:04.68
Symbol table sort                        0    00:00:03.04    00:00:11.40
Pass 2                                 408    00:00:04.88    00:00:18.03
Symbol table output                      1    00:00:00.17    00:00:00.18
Psect synopsis output                    0    00:00:00.01    00:00:00.01
Cross-reference output                   0    00:00:00.00    00:00:00.00
Assembler run totals                  1230    00:00:28.19    00:01:37.56

The working set limit was 2400 pages.
168427 bytes (329 pages) of virtual memory were used to buffer the intermediate code.
There were 150 pages of symbol table space allocated to hold 2706 non-local and 149 local symbols.
2318 source lines were read in Pass 1, producing 23 object records in Pass 2.
60 pages of virtual memory were used to define 57 macros.

                                              +--------------------------+
                                              ! Macro library statistics !
                                              +--------------------------+

Macro library name                             Macros defined
------------------                             --------------
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                             26
_$255$DUA28:[SYSLIB]STARLET.MLB;2                          11
TOTALS (all libraries)                                     37

3086 GETS were required to define 37 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:TTYFDT/OBJ=OBJ$:TTYFDT MSRC$:TTYFDT/UPDATE=(ENH$:TTYFDT)+EXECML$/LIB
```

0403 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



TTYFDT
LIS

TTYCHARO
LIS

TTYCHARI
LIS

TTYDRVDAT
LIS

0404 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY